Location-aware Real Time Strategy Games
Master Thesis
Winter Semester 2012 - Summer Semester 2013

Vadim Costache

April 25, 2013

Contents

1 Introduction: Real Time Strategy games

The purpose of this paper is to describe the development of a new type of GPS-enabled mobile game, a Real Time Strategy / Shooter hybrid. The motivation for this came up during the search for GPS-enabled multiplayer game concepts.
Ever since the first operating system for handheld devices, the spreading of smartphones has intensifed every year.In 2012, the number of smartphones in the world has reached one billion, according to Bloomberg(1)

The rapid expansion of mobile computing presents new challenges and opportunities for both the user and the developer. The ease of use and the presence of touchscreens, GPS receivers, gyroscopes, accelerometers and, since recently, even barometers gives way to new approaches in developing games.

In this paper, we will define a number of computer game genres, classify some of the most popular location-aware games and game platforms and present the evolution of the concept and development of a game type prototype. This game of a genre that has a long legacy among computer games, but is not yet known to the location-aware mobile context. The new genre is that of 'Real Time Strategy'(RTS)(2) games. We will do that with a subgenre of RTS, called Real Time Tactics(RTT)(3). For reasons that will be presented throughout this paper, the type of game can be considered an RTT / Shooter hybrid.

1.1 What is Real Time Strategy?

RTS games are defined as real-time (continuous time) competitive games, in which several players fight against each other, either in a skirmish or team versus team. The purpose is to defeat the opponents by taking real-time decisions on managing resources and troops. The focus is, therefore, on three key elements: **managing resource gathering**, **building a base to provide troops** and **battle tactics**.(2)

This project will focus on a subgenre of RTS, Real Time Tactics(RTT) - which, instead of managing all three aspects of RTS, focuses on **battle tactics**.

Because it is essentially a simplified approach to RTS(3), RTT can provide a good proof of concept and at the same time simple and enjoyable playability. The advantage of having a RTT GPS-enabled game versus RTS is that it greatly reduces the play time, requires less skills and has the potential of being less stressful than RTS.

1.2 Real Time Strateg versus Turn Based Strategy

Turn Based Strategy games(4)(such as chess and board games, for example) allow the player to take his time and plan every move. Implicitly, the duration of the game is greater. This type of game is not in the scope of this project, yet it deserves mentioning, as there are many such games for mobile devices. Some notable implementations are the ones of Scotland Yard(Ravensburger GMBH), Catan(Catan GMBH) and Monopoly(Hasbro, Inc.).

In particular, there are two implementations for Scotland Yard upon which a comparison has been made on whether it is better to port a board game in a real-time(continuous-time) or turn-based mobile GPS-based game(5). These two implementations are Mobilis XHunt(turn based) and MisterX Mobile(real-time). The comparison has been made on 10 aspects: Fun, Smooth Progression, Dynamic Gameplay, Ease of play, Stressless Gameplay, Communication, Strategy, Clear Rules, Low Risk, Education. The conclusion was that there was no favorite between the two, but it has been concluded that these 10 aspects have different weights for the player and that Fun, Smooth Progression and Dynamic Gameplay have a higher individual weight to players than the rest.(5, p. 5)

Based on the knowledge gathered from the above-mentioned comparison, the aim of this project will be to maximize the interactivity and dynamics of the RTS game.

2 Location-Based Augmented Reality Games

Ever since the first operating system for handheld devices, the spreading of smartphones has intensified every year.

The rapid expansion of mobile computing presents new challenges and opportunities for both the user and the developer. The ease of use and the presence of touchscreens, GPS receivers, gyroscopes, accelerometers and, since recently, even barometers gives way to new approaches in developing games.

Although it is one of the oldest additions to the smartphone, the GPS-enabled smartphone is still not ubiquitous. As of now, each company's flagship and most of their mid-level smartphones are GPS-enabled. This makes way to the propagation of GPS gaming. Although the concept is old, very few attempts have been made in this direction and this branch of game development may be considered to still be in one of its early stages of maturity.

This paper is a proposal for extending the genre of mobile real-time multiplayer location-based games.

2.1 Location-based games

Location-based games take advantage of the mobile devices' built-in receivers for global positioning. They provide the user's location with an accuracy ranging from a few to a couple dozen meters. Because the most mobile devices in the world today rely on the Global Positioning System(only recently support for GLONASS has been added to smartphones), we can use the term GPS-games.

GPS-games came up long before this feature has become ubiquitous in mobile phones and tablets. One of the first widespread GPS-games is Geocaching. It is composed of two parts:

a. Placing physical caches at various locations that can be considered interesting or worth visiting and publishing their GPS positions (eg. on websites).

b. Searching for various caches by using a GPS device.

Along with the evolution of smartphones came that of the mobile games. GPS games come in a lot of flavors, from GPS-based tours, adventure and investigation games to various race games - single and multi player and massively multiplayer online games.

2.2 Types of GPS-based games

There is a number of GPS-enabled games and game authoring tools that are available for iOS and Android devices. The ones studied for this paper are : ARIS, Tourality, Wherigo, conTAGion, Shadow Cities, SCVNGR, Please Stay Calm, Parallel Mafia, Parallel Kingdom, Tripventure, Warfinger, Totem, Portal Hunt, aMazing, Ingress, MobileWar, Mister X Mobile, Mobilis XHunt, Own This World, MapAttack.

For better understanding the classification done below, we will first define each type of game:

a. **Adventure/Investigation Game** - Game in which the player plays the role of a character in a story. The primordial characteristics of this genre is that it is focused on immersion in the story, puzzle solving and investigation, rather than on physical skills. Also, the tendency of this genre is towards single player experience, though occasionally multiplayer is also implemented (eg. the ARIS-based game 'Mentira').

b. **Massively Multiplayer Online Game** - This type of game is designed to support large numbers of players (even in the number of millions in some cases) that play and interact in a persistent virtual world. This type of game allows both cooperative and competitive gameplay and is exclusively based on multiplayer. Subgenres include MMO Role Playing Games and MMO Shooters.

c. **Casual Game** - Analogous to the MMO, the Casual Game is targeted at mass at a mass audience and can incorporate any type of game type. The particularity of this genre is that it aims at having simple rules, simple gameplay and requiring no specialized skills.

d. **Racing Game** - It's a genre defining a broad range of games. In the case of computer games, it describes mostly motorized vehicle racing. In the mobile context, it mostly describes racing on foot against time or through a number of checkpoints.

e. **Shooter Game** - This one's a subgenre of action games. It focuses on first or third person experience, speed, aiming and reaction time. Usually the weapon is ranged, although close-combat weapons are included in most games.

We will now classify the games/platforms based on the genres they best fall in:

a. **Adventure/Investigation Games**

   (a) ARIS
   (b) Wherigo
   (c) Tripventure
   (d) Tidy City

b. **Massively Multiplayer Online Games**

   (a) Shadow Cities
   (b) Please Stay Calm
   (c) conTAGion
   (d) Parallel Mafia
   (e) Parallel Kingdom
   (f) Portal Hunt
   (g) Ingress

c. **Casual Games**

   (a) SCVNGR
   (b) Warfinger
   (c) aMazing
   (d) Own This World
   (e) MapAttack

d. **Racing Games**

   (a) Tourality

e. **Shooter Games**

   (a) MobileWar

A special category is represented by Mister X Mobile and Mobilis XHunt, which both bring a board game (Scotland Yard) to the mobile environment. While the former adapts the board game to real-time gameplay(placing it closer to the 'Multiplayer Racing Games', with elements from 'Real Time Strategy Games'), the latter falls in the definition of 'Turn Based Strategy' games.

3 Real Time Strategy Games

This proposal is for the research and development of GPS-enabled Real Time Strategy games. They are to be augmented reality games for single player or multiplayer competitive 'free for all'/'skirmish' and 'team versus team' games. This category of games offers opportunities to also enhance the experience for all the previously existing types of GPS-enabled mobile games.
For this project, the proposed games are :

a. **Territory Takeover**

b. **The War Game**

1. The **Territory Takeover** game is a multiplayer, team versus team competitive game. The players or game author define an area of play, which will be divided into multiple divisions. Each division will be marked by a 'flag' (a GPS marker). To capture the area division, a team must capture its flag. The game ends when all flags have been captured and the winner is the team with most captured flags. Each flag may be given a time that a player must spend next to it in order to capture it. Once a flag (and implicitly the territory) is captured, it remains so until the end of the game. The winner can be decided on flag counting or, alternatively, each flag may receive a number of points, according to the size of the territory marked by it and the difficulty of the terrain.

This game can be enhanced with the use of virtual tools or weapons. For the purpose of this project, the following tools/weapons have been considered :

a. The **Immobilizer** is an ability that can be used by each player to block an opponent from moving. The 'attacker' 'activates' the ability and a circle around him is drawn to show the range in which he can shoot. If an opponent enters the range area, the 'attacker' will select him on the map and shoot. The 'victim' will receive a notification that he is immobilized. A circle or rectangle will be defined around him and he will not be allowed to move outside of it for a given time, say 30 seconds or 1 minute. If he does, he gets disqualified and kicked out of the game. An alternate solution would be that the team loses points, for the case that this is the scoring methodology implemented.

b. **Demobilizer** is an ability that an immobilized player can use. For this project, it will only work on the person that uses the ability. The effect is that a person that is immobilized gets the waiting time halved.

Both the abilities have a common cooldown timer. That means that if a player immobilizes somebody and is immediately immoblized himself, he won't be able to use the demobilizer because of the cooldown following the usage of the immobilizer.

2. The War Game is inspired by Real Time Strategy Games and Airsoft. Two teams are formed. An area of play is delimited on the map. Each player can choose between a number of characters. For the purpose of this project, four characters are proposed: Defender, Marine, Sniper and Heavy Trooper. Each of the four characters has special abilities and characteristics :

a. The **Defender** has the ability to generate shields for short periods of time. Members of the team can hide behind those shields for defence. The Defender may also act as a Medic and heal or revive members of the team. He has low health, long ability cooldowns and a sidearm with short range, small damage and fast cooldowns .

b. The **Marine** has a weapon that can shoot a medium range with medium damage and fast cooldowns. He has medium health.

c. The **Sniper** has two weapons : the sniper rifle that can shoot at distant ranges and deal large damage to single targets and the sidearm, which is the same as the Defender's. His health is low, just like the Defender's. The sniper shot may penetrate the Defender's shield and cause reduced damage to one target.

d. The **Heavy Trooper** has three weapons : the bazooka, the sidearm and mines. The bazooka is a mid-range weapon with splash damage - it therefore can be fired against compact groups, such as the ones that might be hiding behind a shield. The bazooka cannot deal damage through the shield, but it may be shot next to it, causing damage from the side. The damage to each target varies from moderate to small, depending how far they are from the center of the 'projectile explosion'. The mines can be placed randomly on the map and their 'explosions' will not affect the members of the Heavy Trooper's team. Also they cannot be triggered by the members of his team. The damage dealt will be moderate, with splash damage, just like the bazooka projectile. The bazooka and the mines have long cooldowns, therefore the sidearm is added. The Heavy Trooper has high health.

This game has some advantages and drawbacks:
Advantages: It does not require specialized gear and setting, nor does it need long amounts of time to be played. It can be enjoyed with a bunch of friends on a sunny weekend afternoon.
Drawbacks: It highly depends on GPS accuracy. This issue may affect gameplay. It also does not feel as 'real' as real-life games, nor PC games.

4 Requirements

The two above-mentioned games will use a common framework that will enable multiplayer interaction for both 'free for all'/'skirmish' and 'team versus team' approaches. They will require a server to centralize player information such as GPS data and the virtual 'health' attribute. Because the games proposed are fast-paced, they require quick response times from the server, client and and the communication protocol between them.

4.1 Client

The following technologies have been considered for developing the frontend for the application : Android, XCode(for Apple iOS) and multiplatoform APIs such as PhoneGap/Cordova and Titanium. The multiplatform APIs offer the benefit of the 'code once, deploy everywhere' philosophy, at the cost of performance(6)(7)(8). In this case, the approach of using a platform-agnostic framework is disadvantageous. Therefore, the choice of native app development makes more sense for this situation.

The client will be developed with Android, due to its accessibility and versatility and the fact that it's ubiquitous and open source.

4.2 Server

The games that are to be implemented will actively rely on communication with a centralized server. This implies a constant Internet connection on each mobile device. Also, they require fast, low latency message exchanges. The server should be able to handle this for players in the numbers of a few dozens.

The server will be written using NodeJS(9). NodeJS is a javascript framework based on the Google V8 Engine that is designed for web server functionality. Yet, due to the fact that it is a framework and not a standalone server, one can take advantage of the 'Event Loop' functionality to write various other types of servers. This particular feat is beneficial to the purpose of this project. It is also faster than Apache for general purpose, fast message exchange scenarios(10).

4.3 Communication Protocols

During a game, active communication must be performed between the mobile devices and the server in order to multicast all player positions and intentions to all players. There are two aspects to this approach:

a. The game is created on the server, in a so-called 'Lobby' which all the players join.

b. The game is created on the server and once the game starts, all players send their positions to the server. The server is then responsible to multicasting the positions to all the players.

This is, of course, a subset of the tasks that must be performed by the server. There will be other information that must be exchanged periodically, such as the player's health and quantities of various other attributes that will be added during development and/or proposed during trying out the game. In the general idea of communication protocol usage, the list of choices has been narrowed down to three:

a. WebRTC - A protocol that is to be part of the HTML 5 standard. It will be based on the RTP(Reliable Transport Protocol), which is the base for VoIP protocols and is itself based on UDP. It promises to be a very fast standard protocol, appropriate for audio and video streaming and massive multiplayer games.(11) It is still in draft format and there are no official implementations for it. Implementing the protocol itself is outside the scope of this project.

b. WebSockets - A protocol that is part of the HTML 5 standard. It is a low latency TCP-based protocol that promisses to replace http in several types of web applications.(12)

c. RTP - The protocol on which VoIP and WebRTC are based. It is based on UDP and it is designed for real-time streaming of data.(13)

From these three protocols, a choice will be made between WebSockets and RTP. WebRTC is to be left as an option until its official release and implementation for both NodeJS and Android.

4.4 Game Elements

Both games share a number of common features:

a. a number of players that are visible on the map

b. a number of teams, represented by different colors(in the case of free-for-all(skirmish) play, the number of teams equals the number of players)

c. a number of items that each player can use

d. item properties, such as range and cooldown times

e. player health

The 'War Game' also introduces a number of characters to the game. Each character type has a different health value and different tools, with specific properties.

The server must provide a 'game lobby', to which the players connect and prepare for starting the game. This includes character choice for the 'War Game' and a 'Ready' button. When all players declare themselves to be 'ready', a countdown starts and then the game begins.

5 Implementation

Implementation will mean developing a server and a client application from scratch, covering all the functionality needed for the games to work according to the description in section 'Real Time Strategy Games'.

5.1 Schedule

This project, consisting of one server and one client application, is to be developed in four steps:

a. **Development** - During the first iteration of development, the most basic features of the game are to be implemented: basic server functionality that would allow the game to work, basic client functionality and the 'War Game' without all features.(2 months)

b. **Testing and Evaluation** - During this phase, the game and its functionality will be live-tested for feasibility and quality. New ideas will be sought and documented. Most importantly, player feedback will be gathered.(1 month)

c. **Development** - During the second iteration of development, the 'War Game' will be completed and, using its framework, the 'Territory Takeover' game will be implemented. Bugs will be removed and tweaks will be made to the framework and the game concepts to match the player feedback.(2 months)

d. **Evaluation and Completion** - During the second evaluation phase, both games will be tested for playability, player feedback will be gathered and the Dissertation Paper will be completed.(1 month)

5.2 Authoring Tool

As it is not the focus of this project, the Authoring Tool will comprise the most basic elements necessary for setting up the game(such as defining obstacles and probably the border of the gameplay area). Future work might include modifying and/or adding new character attributes, multiple obstacle types and probably extending the RTT games to full-fledged RTS.

6 Documentation on the Games

Extensive documentation has been found from research done on education-oriented GPS-games.(14)(15)(16)(17)(18)(19), describing concepts and approaches in developing platforms and games to this end(16), porting them to different locations(17), comparing them and discussing their functionality(19)(14)(15) or discussing their effect and usefulness(14). Unfortunately for the direction of this particular project, this documentation focuses exclusively on GPS-enabled Adventure Games.

The lack of documentation for the other games, except their websites has left trying them out one by one as the only option to understand their components and functionality. Additional information was retrieved from video descriptions, examples and reviews of the games.

During the preparation of this proposal paper no games or documentation have been found on GPS-enabled Real Time Strategy games. From the searches performed, no location-aware mobile games have been found to fall in the genre of RTS. However, one has been found in the genre of 'Shooter Games' - MobileWar.

The purpose of this project is to create a framework on which the GPS-enabled Real Time Strategy games 'Territory Takeover' and 'The War Game' will be implemented as proof of concept. This framework is to be extended during future work. The names of the games themselves are chosen only by their descriptive nature and are prone to change during the steps of development and evaluation.

7 Application of the Games

The two games are proposed with group teambuilding and recreation in mind. Both games require team strategy and cooperation. Territory Takeover allows for both team versus team and free for all gameplay, allowing for both small and large groups to play. The War Game is to be a fast paced game spanning a time interval in the range of a few tens of minutes. Although it is a RTS Game and not a Shooter Game, the 'War Game' can be seen as a more casual approach to Airsoft. Where it lacks the realism of Airsoft or the immersion of classical computer Real Time Strategy games, it gains in the intensity of real-life experience and teamwork, without requiring specialized equipment(Airsoft) or highly developed skills(computer Real Time Strategy games).

8 The development of the game

This section will follow the development of the application. The backend, frontend and mechanics of the game will be analyzed and presented in their evolution.

The game is composed of three parts :

a. **Mobile App Frontend** - A set of menus for setting up the game and a GUI on top of Google Maps for the game itself.

b. **Mobile App Backend** - The core of the app. This includes the modules for communicating with the server, for in-game protocols and mechanics.

c. **Server** - All players will communicate with each other through a central server that can only host one game at a time.

8.1 Initial plan

The original plan was to develop and Android and iOS app, using a development framework such as PhoneGap. This would assume Javascript / JQuery programming for the UI and application backend. The communication was planned to be established through Websockets. Therefore, for optimum performance, NodeJS was to be used on the server side. For the location presentation, the Google Maps API was to be used.

8.2 Unified vs. Native Frameworks

Once an idea of how the game should look like was crystalized, I started looking for unified frameworks for cross-platform mobile development. The first choice was PhoneGap, which I have used in the Agile Lab in the previous summer and was a bit familiar. The other frameworks that I considered were Titanium SDK, Sencha and Corona. The discussion on which one is better can be resumed to the conclusion that none of them is appropriate for the development of such a game. And here are the reasons :

a. The game is fast-paced and will require a fast backend as well as a frontend that is as fast as possible. The cross-platform frameworks essentially interface only some of the native functionality and display the app through a webview - requiring Javascript or a Javascript-based framework for creating the interfaces. I have tested some these frameworks on top of Phonegap, during the Agile Lab. The purpose was then to find the fastest solution for static menus and slow-paced UIs. Even then, the outcome has been disappointing. Qooxdoo, ExtJs and JQuery were tried out. Qooxdoo performs faster than the other two, but makes it very slow and tedious to develop a complex UI. Still, it is slow for the purpose and feels nonnative. After reading a few articles that compare Phonegap, Titanium, Sencha and Corona, I have concluded that with or without various advantages and disadvantages between them, they are all similar in performance - and therefore too slow for the development of this game.

b. Being a game prototype, all the details on which features of the phone or of the operating system will be used were unclear. I found it unwise to develop on a platform that only permits a number of features that are common to all platforms.

c. Developing native code can prove to be easier, as for example the Android development community is much larger than that of Phonegap developers. The support is both more intensive and extensive for native platforms.

From this point on, I gave up the unified frameworks and had to decide between iOS and Android development. This was an easy choice : Android develpment is free, while iOS development requires a paid developer account and XCode runing on MacOS exclusively. Therefore, I chose Android.

8.3 Communication Protocols

Initially, I have planned to use Websockets for the client-server communication. The reasoning behind it has two main arguments :

a. Websockets is an HTML5 protocol currently presented in draft by the IETF. This protocol provides reliable two-way communication between a server and a client and manages various complex issues or network features that come above TCP. This future standard is developed to replace HTTP and add functionality for technical aspects that HTTP was not covering. The reason for using Websockets for the client-server communication was that it promises to abstract a lot of protocol management issues, while offering speeds close to plain TCP. Also, for the game to run properly, UDP and its child protocols cannot be used - total reliability is required in the communication.

b. Websockets communication can be implemented in NodeJS, which is a framework well suited for fast-paced message exchanges and which has been proven more scalable than, for example, Apache Tomcat.

As I decided to develop the server in NodeJS, the first choice was to use the Socket.IO Websocket plugin. For the client side, I chose Autobahn for Android. The only alternative for Autobahn was not free. After developing a basic Websockets client with Autobahn for Android, I realized that they do not communicate. After a lot of searching, I found out that the protocol draft version that Socket.IO is using is different than that used in Autobahn, and unlike Autobahn, Socket.IO uses an HTTP handshake for establishing the connection. The Websocket-Node and ws NodeJS libraries for Websocket communication were tried out afterwards. Neither were compatible with Autobahn for Android. Then, I have switched to Python and after some technical difficulties both in the Autobahn libraries for Android and establishing the communication between client and server, I decided to give up Websockets and start off with pure TCP. Because writing code in two different languages feels slower, I have switched to Java.

8.4 Maps API

The first version of the app used Google Maps API V1. I chose it because it has more extensive developer support. Unfortunately, the Google Maps API and the Android Support Library cannot work together at the same time, because they need to subclass different Fragment classes.

8.4.1 Map Overlays

The Google Maps V1 API supports the use of Overlay objects to draw on top ofthe map. Most tutotials found online make use of the so-called ItemizedOverlay, which enables easy integration of multiple markers. I have attempted to use this, but here are my conclusions :

a. The ItemizedOverlay uses features that we may call 'magic' - the programmer is kept at arm's distance from understanding exactly what they do. The ItemizedOverlay object uses an ArrayList for storing the map markers as OverlayItem objects. It also needs a function that gives it the size of the ArrayList.

b. Then, by unknown means, the markers are redrawn onDraw. This has proven very difficult to work with. For starters, there is no control over the draw process. Then, the OverlayItem objects cannot be stored in another data structure, such as a HashMap - which I ended up using.

c. Because of its design, the ItemizedOverlay is fast, but useless for the purpose of this app. I chose to replace it with a custom Overlay.

As a replacement for the ItemizedOverlay, I have started to work on how to create a custom Overlay that would fit my needs. I realized how simple it is to modify the onDraw() method and control the display of the markers. This has also allowed me to change their Drawables(icons) dynamically, according to my needs. I have added functionality to change the Drawable for a selected marker or for the marker of a dead character.

A real challenge was to add proper onTap() functionality for the custom Overlay. The advantage of the already-given-up ItemizedOverlay was that it handled marker touch events. The new, custom overlay had no such thing implemented. What I have done was to get the pixel resolution of the screen, along with pixel density data from the system and consider a 0.2 inch radius around the center of the touch on the screen (this is what I have estimated to be a circle to describe the tip of an average human adult finger). That 0.2 inch radius in pixels has been translated in a radius, in meters, on the Google Map. All markers inside that radius were considered and the closest to the center of the touch was chosen. This made choosing a marker out of both crowded and loose situations relatively easy and natural. The tutorial for this will be added as an annex to the paper.

9 ANNEX A - game and game platform websites

ARIS Games: http://arisgames.org/

Tourality - http://www.tourality.com/

Wherigo - http://www.wherigo.com/

conTAGion - http://www.2clams.com/

Shadow Cities - http://www.shadowcities.com/

SCVNGR - http://www.scvngr.com/

Please Stay Calm - http://pleasestaycalm.com/

Parallel Mafia - http://www.parallelmafia.com/

Parallel Kingdom - http://www.parallelkingdom.com/

Tripventure - http://www.tripventure.net/tripventure/

Warfinger GPS - http://www.warfingergps.com/

Totem - http://www.fit.fraunhofer.de/en/fb/cscw/projects/totem.html

Portal Hunt - http://www.totem-games.org/?q=portalhunt

aMazing - http://www.totem-games.org/?q=aMazing

Ingress - http://www.ingress.com/

Mobile War - http://www.mobilewar.org/index.php/en/

Mister X Mobile - http://qeevee.com/projects/misterx

Mobilis XHunt - https://github.com/danielschuster/mobilis/wiki/Mobilis-XHunt

Own This World - http://www.ownthisworld.com/

MapAttack - http://mapattack.org/

10 ANNEX B - game genre definitions

Real Time Strategy Game http://encyclopedia2.thefreedictionary.com/Real-time+strategy+game

> A type of video game in which players exercise strategy along the way, typically to conquer enemies and reach a final destination without being eradicated. For example, to win, players decide which routes to take, what needs to be done and how to do it.

Real Time Tactics Game http://encyclopedia.thefreedictionary.com/Real-time+tactics

> Real-time tactics or RTT is a subgenre of tactical wargames played in real-time simulating the considerations and circumstances of operational warfare and military tactics. It is differentiated from real-time strategy gameplay by the lack of resource micromanagement and base or unit building, as well as the greater importance of individual units and a focus on complex battlefield tactics.

Massively Multiplayer Online Game http://encyclopedia.thefreedictionary.com/Massively+Multiplayer+Online

> A massively multiplayer online game (also called MMO) is a multiplayer video game which is capable of supporting hundreds or thousands of players simultaneously. By necessity, they are played on the Internet, and feature at least one persistent world.

Adventure Game http://encyclopedia.thefreedictionary.com/adventure+game

> An adventure game is a computer-based game in which the player assumes the role of protagonist in an interactive story driven by exploration and puzzle-solving instead of physical challenge.[1] The genre's focus on story allows it to draw heavily from other narrative-based media such as literature and film, encompassing a wide variety of literary genres. Nearly all adventure games are designed for a single player, since this emphasis on story and character makes multi-player design difficult.

Casual Game http://encyclopedia.thefreedictionary.com/casual+game

> A casual game is a video game or online game targeted at or used by a mass audience of casual gamers. Casual games can have any type of gameplay, and fit in any genre. They are typically distinguished by their simple rules and lack of commitment required in contrast to more complex hardcore games.[1] They require no long-term time commitment or special skills to play[. . .]

Racing Game http://encyclopedia.thefreedictionary.com/Racing+game

> One of the more common uses of the term racing game is to describe a genre of computer and video games. Racing games are either in the first or third person perspective. They may be based on anything from real-world racing leagues to entirely fantastical settings, and feature any type of land, air, or sea vehicles. In general, they can be distributed along a spectrum anywhere between hardcore simulations, and simpler arcade racing games.

Shooter Game http://encyclopedia.thefreedictionary.com/Shooter+game

> Shooter games are a subgenre of action game, which often test the player's speed and reaction time. Because shooters make up the majority of action games, it is a fairly wide subgenre. It includes many subgenres that have the commonality of focusing "on the actions of the avatar using some sort of weapon. Usually this weapon is a gun, or some other long-range weapon".A common resource found in many shooter games is ammunition. Most commonly, the purpose of a shooter game is to shoot opponents and proceed through missions without dying yourself.

Turn Based Strategy Game http://encyclopedia.thefreedictionary.com/Turn+based+strategy

> A turn-based strategy (TBS) game is a strategy game (usually some type of wargame, especially a strategic-level wargame) where players take turns when playing. This is distinguished from real time strategy where all players play simultaneously.

Location Based Game (Location-enabled Game http://encyclopedia.thefreedictionary.com/location+based+game

> A location-based game (or location-enabled game) is one in which the game play somehow evolves and progresses via a player's location. Thus, location-based games almost always support some kind of localization technology, for example by using satellite positioning like GPS. "Urban gaming" or "Street Games" are typically multi-player location-based games played out on city streets and built up urban environments.

11 Bibliography

References

[1] J. Yang, "**Smartphones in Use Surpass 1 Billion, Will Double by 2015**," October 2012.

[2] C. Janssen, "**Real-Time Strategy (RTS)**."

[3] J. Eldridge, "**What are Real-time tactics games? RTT games, definition and meaning**," March 2012.

[4] S. Johnson, "**Analysis: Turn-Based Versus Real-Time**," November 2009.

[5] Pascal Bihler, Holger Mügge, Daniel Schuster, Danny Kiefner, Robert Lübke, and Thomas Springer, "**Step by Step vs. Catch Me If You Can - On The Benefit of Rounds in Location-based Games**," 2012.

[6] C. Warren, "**The Pros and Cons of Cross-Platform App Design**," *Mahsable.com*, February 2012.

[7] D. Hoang, "**Native vs. Cross-Platform Development: Which is the way to go?**," April 2012.

[8] D. Wehmeier, "**HTML5, Cross-Platform Compiled, Or Native: Choose Wisely In App Development**," October 2012.

[9] D. Software, "**Node.js: Microsofts Web Empire is Coming to an End**," August 2011.

[10] M. Zgadzaj, "**Benchmarking Node.js - basic performance tests against Apache + PHP**," 2010.

[11] Adam Bergkvist, Daniel C. Burnett, Cullen Jennings, and Anant Narayanan, "**WebRTC 1.0: Real-time Communication Between Browsers**," august 2012.

[12] I. Fette and A. Melnikov, "**The WebSocket Protocol (RFC 6455)**," December 2011.

[13] H. Schulzrinne, S. Casner, and R. Frederick, V. Jacobson, "**Real Time Transport Protocol (RFC 3550)**," July 2003.

[14] Kurt D. Squire, Mingfong Jan, James Matthews, Mark Wagler, John Martin, Ben Devane, and Chris Holden, "**Wherever you go, there you are: Place-based Augmented Reality Games for Learning**," 2006.

[15] Eugenio J. Marchiori, Javier Torrente, Ángel del Blanco, Iván Martínez-Ortiz, and Baltasar Fernández-Manjón, "**Extending a Game Authoring Tool for Ubiquitous Education**," 2010.

[16] D. J. Gagnon, "**ARIS - An open source platform for developing mobile learning experiences**," December 2010.

[17] Chinmay Barve, Sanjeet Hajarnis, Devika Karnik, and Mark O. Riedl, "**WeQuest: A Mobile Alternate Reality Gaming Platform and Intelligent End-User Authoring Tool**."

[18] Christopher L. Holden and Julie M. Sykes, "**Leveraging Mobile Games for Placebased Language Learning**," May 2010.

[19] J. M. Mathews, "**Using a studio-based pedagogy to engage students in the design of mobile-based media**," 2010.

[20] Andreas Ritzberger and Stephan A. Drab, "**TeamTags: Domination An AGPS game for mobile phones**," 2004.

[21] Marko Modsching, Ronny Kramer, and Klaus ten Hagen, "**Field trial on GPS Accuracy in a medium size city: The influence of builtup**," 2006.