"A different kind of ***Alternative Reality*** *occurs in the science fiction film The Matrix, in which the human race is unknowingly living in a simulated virtual reality created by intelligent computers to keep them pacified and content while the computers suck their bioelectrical energy (whatever that is). Maybe this is not so farfetched, because many people prefer to spend their time in the simulated reality of websites such as* ***Second Life****. How do we know we are not just characters in a computer-generated soap opera?"*

– The Grand Design,
by Stephen Hawking and Leonard Mlodinow;
page 42.

University of Alberta

The fAARS Platform
For Augmented Alternate Reality Services and Games

by

Lucio Alberto Gutiérrez Gutiérrez

A thesis submitted to the Faculty of Graduate Studies and Research
in partial fulfillment of the requirements for the degree of

Master of Science

Department of Computing Science

©Lucio Alberto Gutiérrez Gutiérrez
Spring 2012
Edmonton, Alberta

To those who have lent me their shoulders for me to be here. To my beautiful wife, my wonderful family, and my great friends!

Abstract

Today, with gaming technology advancing by leaps-and-bounds, we are witnessing the proliferation of games for entertainment and education. Users can easily record their real-world experiences, through affordable smart phones, equipped with accelerometers, GPS receivers, compasses, and cameras. Commodity virtual worlds enable users to socialize, in realistic environments where they simulate real-world activities, or in imaginary worlds where they can play fantasy games. Finally, augmented reality browsers enable users to ‘annotate’ the real world with digital content through their mobile-devices displays. This synergy of technological advances and perceived service opportunities make the design and implementation of Mobile Augmented Alternate Reality Games (MAARGs) a compelling research problem. In this thesis, we present fAARS, an innovative platform that fully exploits these technologies to support the design, development and deployment of a variety of mobile games, including two systematically evaluated games in collaboration with the Psychology Department and the Faculty of Medicine.

.

ACKNOWLEDGEMENTS

I would like to thank many people who helped me during this wonderful journey. First of all, I would like to thank my supervisors Dr. Eleni Stroulia and Dr. Ioanis Nikolaidis for their endless patience and encouragement. They gave me the opportunity to explore the area of pervasive gaming, and guided me throughout the way. I have no words to express my gratitude to them. Thank you for trusting in me for these two years and a half, and for deciding to work with me right from the beginning of this journey. My utmost gratitude goes to them.

I would like to thank Dr. Sarah Forgie from the Faculty of Medicine for her endless help to design, create, and run the game "Outbreak: Safety First". Thanks to her, the final experiments of this thesis were excellent. I also want to thank M.Sc. students Brian Dupuis, Danielle Lubyk and Dr. Marcia Spetch from the Psychology Department for collaborating with me to create the "Human Geometric Orientation" game.

I want to thank Professors Geoffrey Rockwell and Sean Gouglas for their valuable advice on gaming, in the early stages of this journey.

Thank you to my examining committee Dr. Geoffrey Rockwell and Dr. Abram Hindle for their time, effort, and their valuable feedback on this thesis.

I would like to thank all the wonderful and talented people that I collaborated with, and who made this journey a very exciting one! My thanks go to Dr. Patricia Boechler, Dr. Sharla King, Ph.D. student Erik deJong, Dr. Paul Messinger, and Dr. Mike Carbonaro,

Finally but not least, I want to thank my lovely parents for their lifetime support and encouragement. All my sincere thanks go to my beautiful wife Deydree Valdez for her support and encouragement throughout this journey, and for being with me accompanying me in this journey until the end.

.

Table of Contents

List of Tables

List of Figures

Chapter 1: Introduction

Mobile Augmented Alternate Reality Games (MAARGs) are a relatively new, yet increasingly interesting, class of social applications played using mobile devices in outdoor and indoor spaces, and in virtual worlds [1]. Their popularity is increasing among the general public largely due to four main reasons [2].

1. The hardware capabilities of mobile devices are becoming increasingly powerful. Mobile phones are now equipped with high resolution displays with touch-screens, tiny digital video cameras, QWERTY keywords, multiple gigahertz processors, and a variety of sensors such as NFC (Near Field Communication) and GPS (Global Positioning System). These types of mobile phones are the ones that qualify as “smart-phones”, and in the rest of the thesis when we use this term we refer to mobile devices with these capabilities.
2. The cost of smart-phones has plummeted, and carriers have started offering a wider variety of mobile phones and service plans. In fact, they are widely adopted in rural areas where smart-phones represent a better investment compared to laptop and tablet computers, because of the better mobility and battery power that they can offer.
3. The current mobile operating-system battle [3] has led to the development of more robust, versatile, and simpler to use Software Development Kits (SDK). These powerful SDKs now allow anyone to program a wide variety of context-aware applications [4] that were nonexistent only as recently as a decade ago.
4. Smart-phones quietly shift between WiFi, 3G or 4G connections, which ensures users are always connected to the best available network. This particular feature enables context-aware applications to work smoothly all the time.

The above reasons have motivated a wide adoption of smart-phones, and consequently the development of middleware [5, 6] and a new breed of services, namely location-based services [7, 8] that “know” their surroundings and geographical location of the users. Some of these services can be classified as location-based games [7, 8]: a game where play progresses based on the location and actions of the players. Location-based games are a particular type of pervasive games that use the physical space of our entire world as a game board [9]. The gaming experience of these games can be enhanced with augmented reality applications that allow players to see virtual content on top of physical objects through the screen of a smart-phone. This in conjunction with parallel virtual worlds produces an interesting and new class of pervasive games called MAARGs. This is how the study and development of this new class of games has led to the implementation and deployment of a wide range of MAARGs for entertainment and education in different areas.

1.1 The Research Problem

Examining the landscape of games and commenting on pervasive games in particular, Montola [10] argued that they expand on traditional gaming spatially, temporally, and socially. Differing from traditional gaming in that the game’s location is “unclear or unlimited”, its sessions are blended with ordinary life, and its play occupies a strange social dynamic with its players. Unlike traditional games where the players form a well-defined group and play according to explicit rules, participants of pervasive games advance the game with their everyday-life actions, like walking around with "BotFighters[1]" or the "Pokéwalker"^{TM} Nintendo accessory[2] in their pocket. "BotFighters" was one of the first location-based games developed in 2001 that ran on GPS equipped cell-phones. The objective of this game was to find and destroy other players' robots to earn points and upgrade their robots. In the case of "Pokéwalker", this device was built in 2010 as an

[1] BotFighters http://en.wikipedia.org/wiki/BotFighters
[2] Pokéwalker
http://www.nintendo.com/whatsnew/detail/IkqZYLWkrJPUg3l8XT1MYCHUzQuxM0tl

extension of "Pokémon™ HeartGold Version"[3] and "Pokémon SoulSilver Version"[4] games. The objective of this device was to load a game character and carry it around on the streets to earn points, trade game character items, and enable new virtual worlds in the game. All these tasks would make the carried game character stronger and ready to fight other upper level game characters in their respective game. In these and other location-based games, individuals sometimes may not even be aware of their participation. The timeline of pervasive games is also not well defined: players may not even be aware of when the play starts or ends. A game may be inactive for long periods and then suddenly alert players again that it is on. Finally, in some cases, outsiders may become unaware participants as intermediaries between players, thus expanding the social circle of the game players. These types of games are also referred to as “alternate reality games”, which, according to Martin and Chatfield [11] “take the substance of everyday life and weave it into narratives that layer additional meaning, depth, and interaction upon the real world”.

These games exemplify one natural evolution of online social networks, where the social relations among people are reified in the real world and in the game software, and their real-world interactions are mediated by, and advance, the game logic. Moreover, as they are “embedded” in the real world, they offer a natural opportunity for integration with 3D virtual worlds, which can be used to reflect, simulate, inform and control real-world activities [12]. It is exactly the “smart systems across real and virtual worlds” agenda that has motivated our work on “Mobile Augmented Alternate Reality Games”, (MAARGs), which we see as a particular class of pervasive games that are played, at the same time, in the real world and in a virtual world. In our view, MAARGs further expand traditional gaming by being played in multiple worlds in parallel, with the interaction between the players and the worlds being mediated by computational devices,

[3] Pokémon™ HeartGold Version
http://www.nintendo.com/games/detail/SMbDUlAHoYoFKJ0LEG5P8oBjDy8HDY0N
[4] Pokémon SoulSilver Version
http://www.nintendo.com/whatsnew/detail/IkqZYLWkrJPUg3l8XT1MYCHUzQuxM0tl

namely smart-phones.

Lindley first conceptualized MAARGs [1] in 2004 as *Trans-reality Games*. These games can be deployed and played in “natural” neighborhoods, such as a campus, a city, or a broader geographical area. Game players interact with the games (a) in the real world through location-specific clues communicated to them through mobile phones which “augment reality” and are aware of the player’s locations using GPS or QR codes, and (b) in a parallel virtual world, which is the alternate reality that may reflect the real world in some dimensions and extend it in others. According to our study of previous works on MAARGs like "ARQuake" [13], "Human Pac-man" [14], "Can you see me now?" [16], and others [15, 17, 18, 28, 29, 31], the deployment of this type of game is complex due to the following reasons.

1. There are many technical hurdles that developers have to overcome during the integration of different technologies and platforms. Frequently, game developers spend more time fixing technical issues than creating a game story.
2. Game-play modalities of early works on MAARGs are limited. Although early works included virtual worlds as part of the playground of the game, players were only allowed to perform actions and advance the game in the real world. To enable a truly cross-world game-play modality, where players can log in both in the real and virtual worlds and perform actions and advance the game, a much more complex type of cross-world integration is essential.
3. Early works on MAARGs, specifically in the development of platforms, are limited in terms of the game stories, how games can be played, and where they can be played. The temporal, spatial, and social aspects of MAARGs are not fully supported.
4. Recent platforms and frameworks like "TINYLime" [34], "FRAP" [35], "fAR-Play" [50], and others [51, 52, 53, 54, 55] do not offer the possibility to extend their functionality in an easy way. Major internal changes to their implementations have to be performed in order to support further functionality.

1.2 The fAARS Platform and its Contributions

We have designed and developed an extendable software platform consisting of a set of modules that together enable the implementation and deployment of MAARGs, specifically of the virtual treasure hunt and RPG (Role-playing Game) style of games, and their derivations. This release of our software platform is called fAARS (for Augmented Alternate Reality Services), which has been designed and implemented according to the Event-Driven SOA (Service Oriented Architecture) architectural style. The fAARS modules have been developed and integrated taking into account the style of MAARGs mentioned above and their temporal, spatial, and social aspects.

More specifically, this thesis makes the following contributions to the field. ***The first contribution is the fAARS platform itself***, which consists of the following components.

- *A cross-platform native mobile application*. We have developed our own cross-platform native mobile application to take advantage of the sensors integrated to smart-phones. The fAARS mobile application supports indoors and outdoors games and enables players in the virtual world to log in and play the games by using a simulacrum of the smart-phone where the same mobile application can be run.
- *The game engine*. We have designed and implemented a game engine capable of processing game rules and link the real and virtual worlds. It has been implemented as an event-driven machine, which communicates with other internal components to process the game rules.
- *A mechanism for interfacing with external components as observers of the fAARS platform*. We have developed an API (Application Programming Interface) that allows integration of external systems to our platform in order to extend the downstream functionality of the platform.
- *A virtual-worlds extension.* The fAARS platform supports the integration of virtual worlds. Part of the platform has been pre-packaged as a set of tools that

enables the extension of MAARGs to virtual world scenarios that support parallel simulations with NPCs (Non-Player Characters) and real players interacting with each other in the virtual world. This is a truly innovative feature of MAARGs which is only now beginning to be explored, because it has been rather difficult to implement technically.

The second contribution is the games developed, deployed and empirically evaluated on fAARS as a means of validation and evaluation of our platform. So far, we have developed two different MAARGs using the fAARS platform. Early results from our two game-development studies have demonstrated that this platform supports the deployment of a wide variety of games for entertainment and education. The first game was developed to be played in the *Dreaming* modality, because it was fully implemented and played in a virtual world. This game was developed in collaboration with the Department of Psychology at the University of Alberta. It led to the recent publication [19] of some interesting results about geometric orientation by humans. The second game was developed in collaboration with Dr. Sarah Forgie, a faculty member with the Faculty of Medicine at the University of Alberta. This game fully exploits the capabilities of the fAARS platform, because it was developed to be played in multiple game-play modalities. Its development involves three different versions of the same game played in different spatial and temporal dimensions. The purpose of this game is to demonstrate that MAARGs can be used for learning in a fun way, and it also shows the flexibility of our platform in terms of the types of MAARGs that can be developed.

Finally, the third important contribution of this thesis to Computing Science, and specifically to the field of Pervasive Gaming, ***is the lessons we learned regarding the role of fAARS games in education through our "Outbreak: Safety First" study.*** From our experiments performed in collaboration with Dr. Sarah Forgie, we present our findings about the level of enjoyment of different versions of the same game using different game-play modalities, and discussion on whether MAARGs have potential to be used as educational tools based on our preliminary experimental results is presented.

1.3 Thesis Outline

In the rest of this thesis we discuss our work and experience to date. In chapter 2, background and related research on MAARGs is reviewed. In chapter 3, the design and implementation of fAARS and its constituent components are fully described. In chapter 4, we report on our experience and empirical evaluation of the platform, and discuss some of the objectives of two game deployments. Finally, in chapter 5, we conclude with the lesson learned through this work and we discuss our plans for further development of functionality in the fAARS platform and future MAARGs we are planning to support.

Chapter 2: Background and Related Research

Pervasive applications [56, 57], specifically those that run on smart-phones such as mobile games, Google Maps, GPS navigation systems, and context-aware social networks like Foursquare[5], are now intertwined with our everyday lives. These applications are becoming increasingly popular with the general public, bringing a new era of client-server communications and peer-to-peer interactions that were nonexistent only a decade ago. Pervasive applications can blur [10] the spatial, social and temporal boundaries of our daily activities. However, the study of how and when these boundaries are being transformed, and of what other kinds of applications we can deliver, is in its infancy.

A very similar approach to pervasive applications is ubiquitous computing. Although these two terms are frequently used interchangeably, they differ in terms of the human-computer interactions they support. In pervasive applications [56, 57], like the ones that run on smart-phones, the user operates on one computer or device (smart-phone), and generally is aware of his/her interaction with the application. In contrast, ubiquitous applications, like smart condos for health care [67] or smart environments for big enterprises [68], are commonly deployed using miniaturized and wireless technology that is transparently embedded in the environment. In most cases, the human is not aware of his/her interaction with the (set of) computer(s) running in the background, whereas in some cases the user is fully aware of his/her interaction with ubiquitous technology, like using bottles as a simple interface to access digital content [69]. In general, pervasive applications require the development of sophisticated back-end systems whose functionality is currently provided as web services. Software engineers need to identify the key features of current software architectural patterns and develop new ones in order to deploy more advanced web services, in particular location-based services. With the advent of better-equipped smart-phones, location-based

[5] Foursquare https://foursquare.com/

services will be continuously evolving; they should be adaptable to the pervasive environments of the future. Part of this evolution is the "gamification" of location-based services [20], which brings a new set of gaming experiences through the design and deployment of pervasive games.

2.1 Pervasive Games

According to Lindley [21], the area of pervasive games includes a broad range of games that share a common feature and challenge, the integration of ubiquitous technology with physical spaces to provide a gaming experience. The term “pervasive games” refers to a particular type of pervasive-computing applications, that are run and played in physical environments where ubiquitous and mobile devices are transparently embedded and/or directly used by players to advance the game anywhere at anytime.

In general, pervasive games vary in three main aspects [21]:

1. the computer and communication technologies involved to run and play the game.;
2. the optional inclusion of a virtual world; and
3. the configuration of the physical (and possibly virtual) environment where the game takes place, and how it is involved in the game play.

Two of the most cited works [9, 22] in the pervasive games area attempted to identify and provide a classification of different types of games and genres. These works based their classifications only on the technological aspects of these games, and specifically, how the pervasive technology can be used to enhance the gaming experience, and the challenges involved in the integration of such technologies. In these works, there is a lack of a general conceptual framework identifying the foundational characteristics of pervasive games in general. Let us not forget that there are many enjoyable games that do not need any technological support to be fun and entertaining. For instance, we have games such as *Hide and Seek*, *Capture the Flag*, *Tag*, and their multiple derivations. This kind of games can benefit from pervasive computing to possibly make them more interesting and enjoyable, they do not rely on any kind of technology to be playable and fun. Through our work, we have come to appreciate that the way the technology is involved in these types of games does not have to be the most important dimension along which to classify pervasive games.

In our attempt to better understand pervasive games, we found in Montola's work [10] a very interesting conceptual framework, which clearly identifies a set of common spatial, social, and temporal set of features across pervasive gaming. In this work, Montola states that these features can be enhanced with pervasive technology, leading to a classification of pervasive game that we believe can be applied to categorize all the games that have been developed until now. According to Montola, pervasive games can be classified in the following six categories.

1. *Mixed Reality Games*. Games that are played in physical spaces where virtual and physical components co-exist. Most of the work found in the pervasive gaming area belongs to this type of games. Examples of this type of games include "Pac-lan" [58] and "Ghostwire" [59]. "Pac-lan" mixes the computer-based Pacman game with actions generated by players in the real world. The game runs in a centralized server where the game rules are processed. Players use cell-phones to see other player's positions on a map, and through RFID[6] they can interact with other players in order to capture Pacman or the ghosts. Another example is "Ghostwire", which is a ghost hunting game where players use the Nintendo DSi[7] device to see the ghosts through the display of the device on the real world. In order to advance the game, players have to go to specific physical locations to capture the ghosts. The common aspect of these two games is how the game is brought to reality.

2. *Trans-reality Games*. Games that are played in physical and virtual spaces at the same time, and virtual and physical components can co-exist in both

[6] Radio Frequency Identification http://en.wikipedia.org/wiki/Radio-frequency_identification
[7] Nintendo DSi http://nintendodsi.com/

spaces (*Mixed Reality*). Not too much work has been done in this area, this is due to the technical issues and challenges imposed by trying to integrate the physical and virtual spaces as one. Examples of this type of games include "Human-Pacman" [14] and "Can You See Me Now?" [16]. "Human-Pacman" was one of the fist games that integrated virtual worlds that reflected the state of the game. Any actions generated by players in the real world were always reflected in the virtual world. This was one of the first attempts at having a virtual world running in parallel during game play, although players were only able to log in to the game in the real world. Another game is "Can You See Me Now?", which was one of the first games that allowed players to log in to the virtual world to play. The game was about virtual world players competing against real world players, where virtual players had to hide from the virtual representation of the real world players to avoid being caught in the virtual world. The virtual world reflected the same space configuration where the physical game took place. The common aspect of these two games is how the virtual world also reflects the state of the game, and allows players to play the game at some point.

3. *Adaptronic Games*. Games that are based in virtual reality, and react to external changes that occur in the real world such as temperature, or light intensity. Two examples of this type of games are "NBA Live 365" [60] and "Weakness" [61]. "NBA Live 365" uses real world stats produced by the real players in order to simulate a similar performance in the game. In the case of "Weakness", this game runs in a mobile phone, and generates questions that players have to attempt to answer in order to get points in the game. These questions are generated based on the text messages of the mobile phone users. In these two games, we can see how the information or facts of the real world causes the game to behave accordingly.

4. *Crossmedia Games*. Games that are played with different types of pervasive technology such as cell phones and handheld devices, and where the real world facts are used as input and output during game play. This type of games

should not be confused with *Alternate Reality Games*. An example of this type of games is "Epidemic Menace" [25, 26]. This game introduces a set of gaming devices through which the game can be experienced. This game is later presented as part of the early works on MAARGs, because it includes a virtual world through which players can interact with the game. However, the main contribution of this work is to demonstrate how a game can be experienced differently according to the gaming device used to play the game. Furthermore, in order to simulate a few aspects of the game, it uses the current weather information as input data during game play.

5. *Alternate Reality Games*. Games that used the physical space to create the illusion that anything that happens in the game should not be considered a game, thus creating an alternate reality where players act according to hidden clues. Very well-known examples of this type of games include "I Love Bees" [62] and "Year Zero" [63]. Although these two games were deployed for marketing purposes, their narratives were intertwined at some point with the real world lives of the followers who believed that the stories of both games were actually real. Hidden clues in different electronic devices, websites, and voice-recorded messages were part of the elements of these games' narratives and their "this is not a game" experience. More serious games also part of this category include "EVOKE" [64] and "World Without Oil" [65]. These two games propose a story where the problem is to find the solution to potential real-world problems that can affect us all now or in the near future. The objective of these two games is to generate awareness among the public about real world problems and their possible solutions, which, according to Jane McGonigal[8], can be effectively done through alternate reality gaming.

6. *Reality Games*. Games that have an artificial origin, but everything that happens in the game is real and can have consequences on the real life of players. These games fabricate reality and can generate serious ethical and

[8] Jane McGonigal: Gaming can make a better world
http://www.ted.com/talks/jane_mcgonigal_gaming_can_make_a_better_world.html

legal issues. According to Montola, these games may not even be considered as games, when their players are not aware of their participation in the game. Examples of this type of games are the reality TV shows "Survivor"[9] and "The Biggest Loser"[10], which have an artificial origin, but everything that happens in the game can affect the players for real. In these cases, the players know of their participation in the game, but still their decisions in the game can affect other players or external participants.

Some of these terms permeated the research community such as *Alternate Reality Games*, *Crossmedia Games*, or *Mixed Reality Games*. The other terms are not that easy to find in the literature.

2.2 Mobile Augmented Alternate Reality Games (MAARGs)

During our early exploration in the pervasive gaming area, we found that *Trans-reality Games* perfectly matches the type of games that we are aiming to support with our platform. We believe that the gaming experiences that can be provided with this type of games, as well as the challenges imposed by the integration of multiple technologies into a platform are both very interesting research topics from the software engineering point of view. During our research, we run across an interesting work by Lindley [1], published in 2004, where he elaborates on *Trans-reality Games*. In his work he states that "Trans-reality Games are games that take advantage of pervasive, mobile, ubiquitous, location-based and mixed reality technical infrastructures to deliver new modes of game play experience in which the different contexts of staging are integrated within a unified game space that crosses over technical borders"; this definition exactly covers the type of games that can be developed and deployed with fAARS. Although this type of games were studied and defined quite some time ago, the concept has been barely explored. One of the reasons for this is the set of challenges that the integration of different technologies imposes, specifically the integration of the real and virtual

[9] Survivor http://www.cbs.com/shows/survivor/?ttag=tv;survivor
[10] The Biggest Loser http://www.nbc.com/the-biggest-loser/

worlds.

In this thesis, we refer to *Trans-reality Games* as Mobile Augmented Alternate Reality Games (MAARGs) and use the two terms interchangeably. Using our fAARS platform, we can develop MAARGs that are played in the real world using smart-phones, where the game progresses based on their actions, locations, and interactions among players in the physical space. They are extended to virtual worlds where players and Non-player Characters (NPCs) can interact with each other and advance the game based on their virtual interactions and actions. According to Montola [10], *Trans-reality Games* are pervasive games that systematically blur and break the traditional spatial, temporal, and social boundaries of regular desktop video games. They are limited only by the mobile and ubiquitous technologies involved in running and playing these games, and sometimes limited only by the imagination of game creators due to the constant advances in mobile and wearable technology [25].

In a separate work, Lindley[21] provided a more refined classification of *Trans-reality Games.* Based on how the real and virtual spaces are perceived by players during game play, these games can be classified into two types: *Diegetically Monolithic and Diegetically Polymorphous.*

In *diegetically monolithic* games, the entire game world is perceived as a single world by virtue of having continuous time and motion between virtual and physical worlds, both combined with mixed reality (virtual and physical objects can interact with each other). A *diegetically monolithic trans-reality game* has one physical environment and one virtual world, where game objects, events, and actions in either one of the two worlds have the same cross-world effects, and the modes of interaction with the game in both worlds are the same, thus creating the game experience of having one single world during game play.

Diegetically polymorphous games have multiple parallel worlds, each of them with individual space and time constraints, where every world can optionally support mixed reality. A diegetically polymorphous *trans-reality game* has one physical environment and one or more parallel virtual worlds, where game objects, events and actions in any of these worlds can have different cross-world effects, and the modes of interaction with the game in any of these worlds can be different. In one world, players may be allowed to perform certain actions, when in another they are allowed to perform different types of actions, thus creating the game experience of having multiple parallel virtual worlds during game play.

2.2.1 Examples of MAARGs

Starting in 2002, due to the lack of standardization for application development on mobile platforms, early works on MAARGs focused more on resolving technical issues rather than on the game story and content development. Let us take “ARQuake” [13] for instance. This game was one of the first outdoor/indoor augmented reality applications that addressed important technical aspects trying to convert the regular desktop application Quake into a location-aware game in 2002. Cumbersome equipment had to be developed to fit some of the hardware requirements of the game-play. "ARQuake" was deployed outdoors using GPS, and indoors using fiducial markers. Players of this game were able to see the monsters in a first-person view through goggles that displayed the game characters on top of physical objects. This was one of the first attempts at providing an augmented reality experience in a game. In order to shoot the monsters, players had to use a plastic gun through which the direction was detected. Two years later, “Human Pac-man” [14] came out as one of the first outdoor augmented reality applications that blended the real and virtual dimensions, for which cumbersome equipment had to be developed again. Players played the popular game "Pacman" on the streets, while the state of the game was reflected in a virtual world as the game progressed. Players were able to see the characters of the game using augmented reality. In 2005, “Magic Land” [15] was one of the first Augmented Reality Platforms that blended 3D avatars of live human beings with 3D computer-generated animations, fusing the real and the virtual worlds in one and the same environment. Also, between 2004 and 2006, MAARGs, such as “Can you see me now?” [16], “Uncle Roy All Around You” [17], and “I Like Frank” [18], are specific cases of games that make use of mobile devices where physical playgrounds are blurred with virtual worlds, and players in both worlds participate to advance the game. These three games were implemented as a variation of the classic game *Tag*, where players were supposed to find virtual players walking around in a virtual world. Players in the real world were able to see the virtual players on a map rendered in the mobile devices carried by the real-world players. And finally, “NetAttack” [24] and “Epidemic Menace” [25, 26], are both games whose game stories revolve around the virtual treasure hunt game domains, and are also part of the early works that study the technical challenges imposed by the integration of hardware and software technologies in MAARGs development. "NetAttack supports outdoor and indoor game play, which afford different game-play experiences. The game is about collecting items with which one can build a secret password to disable a centralized database of a virtual evil corporation. The indoors game is played using a computer, and the outdoors one using GPS. In the case of "Epidemic Menace", the authors of this game argue that a game can be experienced differently using a variety of devices, and different game capabilities can be enabled using different technologies. The purpose of this game is to allow a team of players to collaborate with each other to avoid a mutated virus to contaminate a university. Players have to find evidence of how the virus was created and where it started. Different technologies in the game enable players to find different clues.

More recently, the integration of virtual worlds started to play a more important role among new MAARGs along with the popularization of smaller devices such as handheld computers and regular cell phones used to play these games. For example, “The Timewarp” [27] is a time-travel game that combines virtual and physical spaces during game play, allowing players to explore the past through a virtual world, while the game runs entirely in the real world. The real world is augmented with media content that is displayed on physical objects, while players have the opportunity to explore deeper into the past by visiting the virtual world trough wearable computing. A more recent MAARG is “Alien Contact!” [28]. This is a detective game where players have to find clues in the real world by solving challenges in the form of questions, and looking for virtual characters in a parallel virtual world using handheld computers. Although in these two works the use of a virtual world is an important component, the spectrum of actions that can be done in the virtual world are still limited.

All these examples of MAARGs, both early and more recent games, share many requirements in common, including continuous communication between the real and physical worlds, augmented reality content, and a notion to extend the game to indoor environments to successfully map the real and virtual space configurations. Recognizing these many commonalities, our long-term objective was to develop a general platform that supports the implementation of MAARGs that are fun to play and also educational; according to Chang et al. [29], pervasive technologies have proven to be helpful tools for teaching and learning. This is a substantial challenge from a software design and engineering perspective. According to our study, the development of MAARGs requires the following supporting technologies:

1. the integration of a variety of hardware (including, but not limited to, desktops, mobile phones, and environment annotations through QR codes, RFIDs and sensors) to enable players to flexibly interact with the game;
2. the development of middleware to connect all these devices and to communicate the game play;
3. the extension of the game play through the integration of virtual worlds as part of the playground; and
4. more importantly, perhaps, the development of engaging and fun game stories and narratives.

In the process of designing and developing a MAARG platform, we have analyzed and compared the software architectures of existing MAARG platforms. At the same time, we have developed our own set of components that are included in our platform in order to facilitate the design, development and deployment of MAARGs by game creators and possibly educators.

2.3 Constituent Technologies of MAARGs

Based on our study of previous works on MAARGs, the common set of constituent hardware and software technologies that support the development and deployment of MAARGs are: 1) mobile devices, 2) augmented reality applications, 3) virtual worlds, and 4) middleware to communicate the game to all players and connect all the components of the game.

2.3.1 Mobile Devices

Cumbersome and expensive wearable equipment of previous MAARGs is now being replaced by powerful, miniaturized, and cheap smart-phones that can provide the same set of hardware functionalities as in the early years of MAARGs [23]. There has been a constant evolution of mobile hardware starting with backpack computers plus head mounted displays, then replacing them with mobile PCs and handheld devices, and finally giving as a result the proliferation of the current smart-phones. The advent of smart-phones [2] has brought the opportunity to game developers to use them as the sensorial interface through which the physical environment can be recognized during game play. They now come with powerful sensors such as GPS (Global Positioning System), magnetometers, tiny video cameras, and NFCs (Near Field Communication). With a GPS receiver, it is possible to track the outdoor position of a device by triangulation of signals coming from GPS satellites. The magnetometer is a sensor that detects the orientation of the device and is frequently used to properly display computer generated images on top of physical objects, among other multiple uses in mobile applications in order to enhance the end-user experience. And an NFC sensor allows smart-phones to establish a two-way radio communication with other smart-phones and devices, with which the peer-to-peer communication can be implemented to share data. Through this set of sensors, indoor and outdoor environments can be recognized and game content can be provided based on the location of players. Besides, the popularity of smart-phones has led to the development of more robust, versatile, and simpler to use SDKs [2], which enables the development of more advanced Graphical User Interfaces (GUIs);

thus providing a better gaming experience to end-users in the front-end.

2.3.2 Augmented Reality

There are many applications that can be supported with current smart-phones, most interesting among them (for our purposes) being the augmented-reality browsers and toolkits. The term *"Augmented Reality"* refers to the annotation of media content on physical objects. This virtual content can be seen through the display of a video camera. In 1994, Milgram's *"Reality-Virtuality continuum"* [30] placed augmented reality on a continuum between the real and purely virtual environments, such as regular computer games. In the early MAARGs, a wide variety of techniques were implemented in order to display virtual content through the use of big and expensive displays. Some of these techniques were very complex, ranging from feature-extraction techniques [31] to a wide variety of hybrid techniques [32, 33] that combine GPS readings with inertial and other sensors. The feature-extraction techniques are useful for mobile phones that have only a GPS receiver. In such cases, samples of real world images obtained with the phone camera are compared against a remote database of images, and in conjunction with GPS readings, the outdoor location of the phone and its facing direction can be calculated. Since this technique is based on image sampling, it can be also used indoors. Hybrid techniques usually combine GPS readings with calibrated inertial sensors (translation + rotation) for calculation of the outdoor location and orientation of the device. More recently, thanks to the advance in hardware and SDKs, there are novel and simpler techniques that use GPS, triangulation of access points and cell towers, and magnetometer readings in order to provide augmented-reality experiences indoors and outdoors. This was a great innovation a few years ago, and now there are many applications and toolkits that allow the create augmented-reality experiences that run on smart-phones such as Layar[11], Junaio[12], ARToolkit[13] (succeeded by Studierstub Tracker[14]), and

[11] Augmented Reality Browser Layar http://layar.com/
[12] Augmented Reality Borwser Junaio http://www.junaio.com/
[13] Software Library ARToolkit http://www.hitl.washington.edu/artoolkit/

Aurasma[15]; which are general purpose augmented-reality applications that enable augmentation of the physical world with virtual content.

2.3.3 Virtual Worlds

In our view, a virtual world can serve as the alternate-reality component in MAARGs. Although in the literature of previous works on MAARGs the use of virtual worlds is mentioned, they do not specify any particular software used in the development of such virtual environments. In early works of MAARGs, virtual worlds supported one-way communication, where physical players were the only ones able to send messages to the virtual world and change the state of players to advance the game. Now with the proliferation of commodity and powerful virtual worlds such as Second Life[16], Open Simulator[17] and Open Wonderland[18], virtual worlds can support a two-way communication, where physical and virtual players can interact with each other in the real and virtual environments. This allows the development of new and more interesting MAARGs by creating dynamic virtual environments with NPCs controlled by external systems. Developers can take advantage of advanced capabilities such as the embedded physics engine, or the new set of web-services that allow sending/receiving data and actions in the virtual world.

2.3.4 Middleware and Platforms

The middleware is the component of MAARGs that is in charge of connecting all the pieces of the game, communicate the game to the players, and processing the game rules. In the early works of MAARGs, the development and deployment of this kind of pervasive game used to be done using an ad-hoc approach. The requirements and rules of every new game were first defined in order to design and implement a suitable middleware to run the game. This was due to the fact

[14] Software Library Studierstub Tracker http://handheldar.icg.tugraz.at/stbtracker.php
[15] Augmented Reality Browser Aurasma http://www.aurasma.com/
[16] Virtual World Engine Second Life http://secondlife.com/
[17] Virtual World Engine Open Simulator http://opensimulator.org/wiki/Main_Page
[18] Virtual World Engine Open Wonderland http://openwonderland.org/

that the area of pervasive gaming was highly innovative at different levels [35]. Currently, new game ideas and mobile platforms are emerging at a higher frequency compared to other interactive systems.

One of the first platforms that allowed the implementation of pervasive gaming was TINYLime [34]. In this work, Mottola et. al. explored the possibility of using sensor devices (or *motes*) as a platform to support pervasive games. The *motes* are spread all over a physical environment in order to provide the experience that a player is immersed in a virtual world. In order to advance the game, players have to interact with the sensors and these interactions are recognized by the system as actions during game play. "Save the Princess!" is a collection game implemented using the TINYLime platform, where a group of players have to find the correct items and ingredients in order to face the black night to save the Princess. The platform supports peer-to-peer interactions among the players through Wi-Fi in order to exchange collected items since only one player can fight the black night at the end, as well as player-to-*motes* interactions in order for the players to be able to pick up the items. In parallel to this work, a number of MAARGs started to appear, and later, a small set of platforms that allow the implementation of this kind of games were created. Although, there is still a lack of formalization about platforms for pervasive games [35], in this thesis, we compare these platforms based on some of the common characteristics found across different MAARGs and platforms and their underlying technologies that support them. We have attempted to categorize this set of commonalities and capabilities found in multiple works in order to provide a comparative table of some of the MAARG platforms developed until now. As seen in table 1, we have based our comparative table on seven dimensions:

1. *Spatial support*. This dimension refers to the spatial aspect of MAARGs, and describes whether a platform supports the deployment of games indoors or outdoors, and if it supports extending games to virtual environments.
2. *Social/Collaboration during game play*. Pervasive games are social by nature, and this dimension refers to whether a platform provides technical support to

enhance the social interaction or collaboration for game completion. For example, if a platform allows and recognizes teams as part of the rules to play a game, the completion of a game is based on the actions of each of the members in a team.

3. *Forms of Augmented Reality*. This dimension refers to any type of support for augmenting the players' reality with virtual content seen through the screen of a smart-phone. In our analysis, we consider this term in its most general definition, namely not limited to simply displaying imagery on top of physical objects, but rather to any ways of augmenting the gaming experience by providing different types of data that can be displayed on a map, or using a GUI through which gaming data can be communicated to players in some form.

4. *Game domain support*. The underlying technologies of the platforms we reviewed allow the deployment of games of a specific game genre, such as virtual-treasure hunt (collection game), RPG (role-playing game), or capture-the-flag (chasing style game).

5. *Communication support*. This dimension refers to the way the communication between the game server and the players is performed. Most of these platforms support only a centralized client-server communication style.

6. *Game-authoring support*. Part of the objectives of having platforms that allow the development of MAARGs, and pervasive games in general, is that game developers do not have to worry about the technical aspects of the game. This can be enhanced by providing an authoring environment for game development, which certainly speeds up the building time of a game.

7. *Game-play modalities*. This dimension refers to whether a platform allows players in the real and virtual worlds to perform the same or a different set of actions to advance the game. Most of the current platforms only allow one game-play modality.

Table 1. A Comparison of Different MAARG Platforms.

	Spatial Support			Social/Collaboration Game Play		Forms of Augmented Reality			Game Domain Support			Communication Support			Game-play modalities	
PLATFORM	Physical Indoors	Physical Outdoors	Virtual Worlds	Individual	Teams	Map	Augmented Browser	Raw Game Data	Virtual Treasure Hunt	RPG	Capture-the-flag	Peer-to-peer	Client-server	Game Authoring	In-world game play	Physical game play
fAR-Play [50]	✓	✓		✓			✓	✓	✓				✓	✓		✓
SCVNGR [51]		✓		✓		✓		✓	✓				✓	✓		✓
Playar [52]	✓	✓		✓		✓	✓	✓	✓	✓			✓	✓		✓
MUPE [53]		✓		✓				✓	✓	✓		✓	✓			✓
Mobile Chase [54]		✓		✓				✓	✓				✓			✓
FRAP [35]		✓			✓	✓		✓	✓		✓		✓			✓
Wherigo [55]		✓		✓		✓		✓	✓				✓	✓		✓
PCAFPEA [66]		✓		✓		✓	✓	✓	✓			✓	✓			✓
fAARS	✓	✓	✓	✓	✓		✓	✓	✓	✓		✓	✓		✓	✓

As seen in Table 1, a total of nine platforms are compared, including fAARS (the last one in the list). Each of these platforms supports the deployment of pervasive games that exploit different characteristics of MAARGs in general. A brief overview of each of these platforms follows.

fAR-Play is a platform that supports games in the virtual treasure hunt style, and allows to include challenges in the form of questions as part of the game play. It is implemented in the client-server architecture style, and supports game play indoors through QR codes and outdoors using GPS. It provides augmented reality content using Layar and maps to locate the treasures, and feedback to players is provided through a mobile web site. The functionality of the platform has been wrapped within a RESTful interface, and an authoring environment has been included in the platform for the rapid prototyping of games.

SCVNGR is a platform that supports games in the virtual treasure hunt style, and supports different kinds of challenges such as taking pictures, decoding QR codes to earn rewards in the game, and answering questions. It supports outdoors game play using GPS, and has been implemented in the client-server architecture style. It supports maps as the augmented reality feature of games, and feedback to players is provided through a native mobile application downloadable for the iPhone and Android phones. It also provides an authoring environment where treasures and challenges can be added to a game.

Playar is a very new open source platform that is about to be released in 2012. From the description on the website of the project, it supports games in the virtual treasure hunt and RPG styles. It supports indoors and outdoors game play using Layar, and feedback to players is provided through a mobile website. The games developed in this platform are supported where Layar is also supported, on the iPhone and Android platforms. Playar has been implemented in the client-server architecture style, and it provides a set of configurable libraries to develop new games.

MUPE is an old open source platform that allows to develop games supported only by mobile phones that can run the J2ME[19] (Java 2 Platform Micro Edition) virtual machine. It supports the deployment of games in the virtual treasure hunt and RPG style, and allows outdoors game play and peer-to-peer player interactions through Bluetooth. This platform has been implemented in the client-server architecture style, and provides a set of configurable files that are used to create games.

Mobile Chase is a platform, which according to its authors, supports the

[19] J2ME http://www.oracle.com/technetwork/java/javame/index.html

development of market-ready games, and provides a toolkit for the rapid prototyping of outdoor game play. The virtual treasure-hunt games developed with this platform are supported only by mobile phones that can run the J2ME virtual machine.

The ***FRAP*** platform supports the development of pervasive games in the virtual treasure hunt and capture-the-flag styles. It supports outdoors game play using GPS, and requires the collaboration among players to advance the games. It was implemented in the client-server architecture style, and uses a map to locate other players during game play as part of the augmented reality feature of games.

Wherigo also supports games in the virtual treasure hunt style. It provides a game authoring application for developing games that can be downloaded to a mobile application that runs on GPS-enabled pocket PCs and Garmin[20] handheld devices. It supports outdoors game play using GPS, and supports maps as part of the augmented reality feature of games.

PCAFPEA (Prototyping a Context-Aware Framework for Pervasive Entertainment Applications) is a prototype framework for outdoors context-aware applications. Although the authors of this work mention the support for peer-to-peer interactions, no details about this feature are included. It allows the deployment of games in the virtual treasure hunt style with the very interesting augmented-reality feature of providing 2D and 3D maps of the space where the game takes place.

Table 1 summarizes the results of our comparative analysis of the currently available platforms for developing and deploying MAARGs, based on the dimensions we discussed above. As seen in Table 1, our fAARS platform distinguishes itself from the other platforms in three dimensions.

[20] Garmin devices http://www.wherigo.com/garmin/default.aspx

1. fAARS is the only platform in the list that supports the deployment of games in a virtual world as an extension of the physical playground where actions and events can also happen. The set of virtual actions and events can be recognized by the fAARS game engine in order to process the corresponding game rules.

2. fAARS supports peer-to-peer interactions among players. This is done in the virtual world, where both real and virtual players can interact with each other and notify the game server about it in order to compute any corresponding game rules. These interactions in the virtual world are not mediated. Objects in the virtual world are capable of sensing these interactions individually.

3. Virtual world game play is supported. This means that any virtual player can advance the game as any regular physical player, and possibly the set of actions that can be performed in the virtual world can be extended, allowing more interesting interactions with the game.

Our main objective with this comparative table is to demonstrate that future platforms can be described and possibly designed in terms of these dimensions. As seen in Table 1, every platform supports different aspects of pervasive gaming, which can possibly help to start formalizing the area of pervasive games.

2.4 MAARGs and their Educational Potential

Pervasive games are seen as a very promising technology for education [22]. According to Chang et. al. [29], pervasive environments, such as pervasive games in a physical space, are a good opportunity to be used as learning scenarios where educational content can be embedded for students to explore and discover. Evidence of using MAARGs for educational purposes can be found in some of the earlier games. For instance, the educational aspect of the “The Timewarp” [27] time-travel game is to provide the opportunity to players to play a game while learning about the history of a city in Germany within the spatial and temporal dimensions. Another example is “Alien Contact!” [28], a game that teaches high-school students the importance of collaboration in order to find the answer of why aliens have landed on Earth. In this game, students also have to solve math and literacy challenges in the form of questions, thus providing the opportunity to students to learn more about different subjects during game play.

Technology is bringing new pedagogical opportunities to the area of education. Embedding educational content in pervasive gaming seems like a good opportunity to help teachers to apply innovative learning tools to teaching, and hopefully students can find them interesting enough to play with. With our MAARG platform we are trying to see if learning is possible through the deployment games and exploring different learning experiences through different game-play modalities. With this approach, we will try to corroborate whether pervasive environments are good options for learning in comparison to regular desktop computer applications. Although the integration of different technologies into a platform to generate MAARGs is a good enough challenge from the software-engineering perspective, another important challenge is to facilitate the creation of games that can be fun and educational.

Chapter 3: Design and Implementation of fAARS

fAARS supports the deployment of games that are played in outdoor environments using GPS, and indoor environments where localization is performed by the players using their smart-phones to read QR codes that are affixed on the surface of artifacts at various points of interest. Games can also be extended by integrating virtual worlds in the game. It is now straightforward to build models of real world buildings in any number of commodity virtual-world engines, like, for example, Second Life and Open Simulator, and to integrate these models in the game play of a game. At this point, we are considering two types of mapping of the virtual and real worlds where a) the virtual worlds augment the real world with more details based on a literal mapping of coordinate spaces between the two worlds, or b) the virtual world represents a conceptually different extension of the real world playground, where specific points of interest are mapped across the real and virtual worlds. These mappings can be specified and packaged as part of the fAARS platform using Open Simulator. The next question becomes the nature of the temporal correspondence of the activities between the game play in the real and virtual world(s). For now, we have envisioned two alternatives. On one hand, events in one world have a direct and immediate impact on all other worlds with time proceeding similarly in both worlds. On the other, it is possible to have time advance faster in the virtual worlds in order to explore the alternative scenarios and inform the players' decisions in the real-world game play. The fAARS platform currently supports both alternatives.

3.1 Requirements

According to Montola [10], pervasive games expand on traditional gaming spatially, temporally, and socially. Spatially expanded games can be played on spaces as small as a building, a park, or a city; or as big as in an entire country, or several countries around the globe. Temporally expanded games can be played for very long periods of time, interlaced with the ordinary lives of players. Or they may run for just a few hours, which request players to play and finish a game with clear and achievable goals. Socially expanded games can involve people outside the game to become active participants in a game, or they can be even unaware of their participation. This can also lead to different type of games, where the challenge is to finish the game in collaboration with teammates geographically distributed in different locations far from each other, or even having games where the main challenge is to avoid contact with other players. These basic properties help to define a variety of MAARGs based on different game-play modalities.

Inspired by Montola's properties of pervasive games in general, and considering the challenges [22, 36] imposed by the development of MAARGs, we have envisioned fAARS as a platform for indoor and outdoor MAARGs, with the following requirements.

1. The platform should support games across different spatial dimensions. Game players can interact with the games a) in the real world through location-specific clues communicated to them using QR codes or augmented reality content by GPS, and b) in a parallel virtual world that may reflect the real world in some dimensions and augment it in others.
2. The platform should support the deployment of games that can run for short and long periods of time. Game players can stay logged in to the game for as long as they game demands it, and they should be able to continue playing even if they are offline for cases when the game runs for a long period of time. In fAARS, this is supported by maintaining the state of the game in a parallel virtual world, aware of offline and online players.
3. The platform should allow social interaction during game play. Players should be able to play and/or collaborate either in the real or the virtual world by allowing different game-play modalities. The platform should support virtual and physical game play.

4. The platform should support games using the capabilities of the typical high-end smart-phones of today, equipped with GPS receivers, accelerometers, electronic compass, and a digital video camera.
5. The platform should facilitate the implementation of MAARGs that include a fun and educational narrative and interaction. These games open the possibilities to the study of educational aspects in different areas of study that MAARGs can bring.
6. The platform should be easily extendable and modifiable to include further downstream functionality in the future.

This blend of requirements and technologies makes the study of this kind of pervasive games unique, along with the significant complex and interesting challenges they impose.

3.2 Game-play modalities and Type of Games

fAARS supports the deployment of MAARGs in the virtual treasure hunt and RPG [37] styles or game domains. The virtual treasure-hunt style is about players exploring around while collecting items in order to earn rewards in the game. The RPG style extends the concept of virtual treasure hunt where players can perform actions according to their role or character in the game. Obviously, the main difference between regular RPGs and the kind of RPGs that fAARS supports is the time/space scenario where these games are played, and how they are played. The RPGs supported by fAARS use the real world space and time as the main game board, and they are played using smart-phones. These two game styles blend with the concept of *Trans-reality Games* in the sense that their game play does not recognize the boundaries of the real and the virtual worlds playgrounds.

These game domains are extended in fAARS by incorporating virtual worlds as the alternate reality. According to Benford et al. [22], bridging the physical and the virtual worlds imposes a considerable level of complexity and challenges at the narrative and implementation levels, but they open new and interesting possibilities for exploration from the commercial and research perspectives, specifically in education. These possibilities include the exploration of different game-play modalities that have not yet been adequately studied. fAARS supports three different game-play modalities [21].

1. *Game play in the real world:* In this modality, game interactions are limited to other players and game objects based on their location in the physical world. Lindley [21] refers to this particular game-play modality as *Dormant*.
2. *Game play in the real world reflected in a virtual world:* In this modality, physical game interactions are reflected to a virtual world. Every physical player has a representational object or avatar in the virtual world during game play. Through their avatars, players can interact with other virtual players and objects in an alternate world. According to Lindley [21], this game-play modality allows the deployment of two game-play modalities that qualify as *Astral Projection* and *Mirror*.
3. *Game play in the virtual world:* In this modality, players use the spatial configuration of a physical space to update their locations in the virtual world, where game interactions with other players and objects are based on their location in the virtual world only. Although this game-play modality allows the deployment of games that can also qualify as *Mirror*, the fAARS platform supports games that qualify as *Dreaming*, which according to Lindley [21], are games where players are logged in to the virtual world using a computer, and all interactions and movements happen in the virtual world.

In each of these game-play modalities, the location of players in the real world is essential. We have extended this notion to virtual worlds, which support the design and implementation of MAARGs that include more interesting narratives and game plays with parallel virtual world simulations and Non-Player Characters (NPCs) in real time.

3.3 An Illustrative Example of a fAARS game

In order to explain the fAARS approach to developing MAARGs, we will describe the capabilities of the platform and the game-development process it supports, using the well-known Pacman game as an illustrative example. This game is an immensely popular arcade game that starred a small, yellow, puck-shaped character being chased around by ghosts while trying to eat every “pill” on a map. The rules of this game are very simple. Pacman must attempt to collect all the “pills” and “powerpills”[21] all around a space, while avoiding being caught by four Ghosts chasing it. If Pacman can eat all the “pills” without being caught, then Pacman wins. Now, let us picture Pacman, but played on a real-life scale: city blocks are the map, Pacman and the Ghosts are players with smart-phones, and QR codes scattered throughout the game area are the “pills” that Pacman needs to scan in order to win the game.

In 2004, graduate students at the New York University created such a version of the game: “Pac-Manhattan” [38]. It involved ten players: one Pacman, four Ghosts, and a “controller” for Pacman and each of the Ghosts, whose responsibility was keeping each player updated on the positions of others out in the field.

Our version of the game, developed in the fAARS platform, is very similar: Pacman and four Ghosts each suit up with a smart-phone equipped with a QR code scanner and a data connection. QR codes are affixed around the game area representing the “pills” that Pacman has to collect, and on the players; no special-purpose controllers are required.

In the following paragraphs, a quick overview of the development and deployment of fAARS-Pacman is provided, and details about its implementation are described in the following sections. In order to create and run fAARS-Pacman and any MAARG in general, a two-step process should be followed: (a) entering the game rules into the database, and (b) uploading the media content that will be used during game play to an online repository.

In order to create the game rules, the *Actors* of the game should be first identified. An *Actor* represents a player in the context of a fAARS game. In this case,

[21] From now on, we will simply use “pills” to refer to both “pills” and “powerpills”.

fAARS-Pacman has a total of five *Actors*, one Pacman and four Ghosts. The rules of the game can then be created, in terms of *Event-Condition-Action (ECA)* rules. When all the ECA rules of a game are defined and entered into the database, the media content that will be used in the *Heads-Up Display* (*HUD*) of the game is uploaded to an online repository. The HUD is the component through which players receive feedback about game state changes during game play, and runs on the smart-phone. Once the game rules and the media resources have been uploaded, five player slots should be created in the database, each of them with a different game role; in this case either a Pacman or a Ghost.

Players can login using the cross-platform mobile application that comes as part of the platform. To login, players should enter a username and a password, which corresponds to a specific *Actor* in the game, so that the fAARS game engine recognizes everyone during game play. As the game progresses, the game rules are evaluated, and actions are executed by the fAARS game engine in real-time as *Events* happen during game play. Players walk around a specific physical area while decoding QR codes affixed to players and walls to advance the game, thus generating real world *Events*. In the context of the fAARS platform, an *Event* describes an interaction between two *Actors*. There can be multiple *Events* during game play, and they are all managed by the fAARS game engine in order to process the *ECA* rules to update the state of the *Actors* in the game. Furthermore, at any time during the game, players can track their scores and status on their smart-phones, and they can continue playing the game until either Pacman dies or when Pacman finishes eating all the "pills".

During a fAARS-Pacman game, players can login to the game and play using their smart-phones. All the interactions among the *Actors* in the game and consequently the *Events* that are generated occur entirely in the real world. This set of interactions among *Actors* in the real world is what qualifies as this game as exhibiting the *Dormant* game-play modality. Although this is the natural way of playing fAARS-Pacman using smart-phones, this game can be played in multiple other game-play modalities as well. For instance, this game can be played in the *Dreaming* modality, which is when the game happens entirely in the virtual world.

Or, if a game developer is looking for more interesting *Actors* interactions, fAARS-Pacman can be also played in the *Astral Projection* modality. In this scenario, the Pacman *Actor* can be walking around in the real world looking for “pills” while being chased by the Ghost *Actors* in the virtual world. To achieve this game-play modality, the Pacman *Actor* has to have a virtual representation or avatar walking around in a parallel virtual world which allows interactions with the Ghost *Actors,* who can be either players logged in to the virtual world or NPCs. When players are logged in a virtual world, they have a simulacrum of their smart-phone, which displays the *HUD* through which players can receive feedback about game state changes.

According to our comparison of MAARG platforms presented in Table 1, fAARS-Pacman could be easily implemented in the *Dormant* game play modality in any of the platforms previously presented. Although each platform supports the deployment of different game styles, this game could be adapted according to the capabilities of any of these platforms. However, although these platforms support the deployment of MAARGs in general, they do not implement one important feature that makes fAARS-Pacman a *Trans-reality Game*, namely the integration of a virtual world where the game play can be extended, allowing players in the real and the virtual world to interact with each other during a game. The fAARS platform supports the implementation of fAARS-Pacman in the *Dreaming* and *Astral Projection* modalities because the fAARS Game Engine can process rules that can be activated by virtual world *Events*.

In sum, the fAARS platform supports multiple game-play modalities without having to change any of the game rules. This allows for a great flexibility of experimentation by having the game being played in multiple worlds in parallel, while players can collaborate to advance the game either in the real or the virtual world.

3.4 The Run-Time Environment

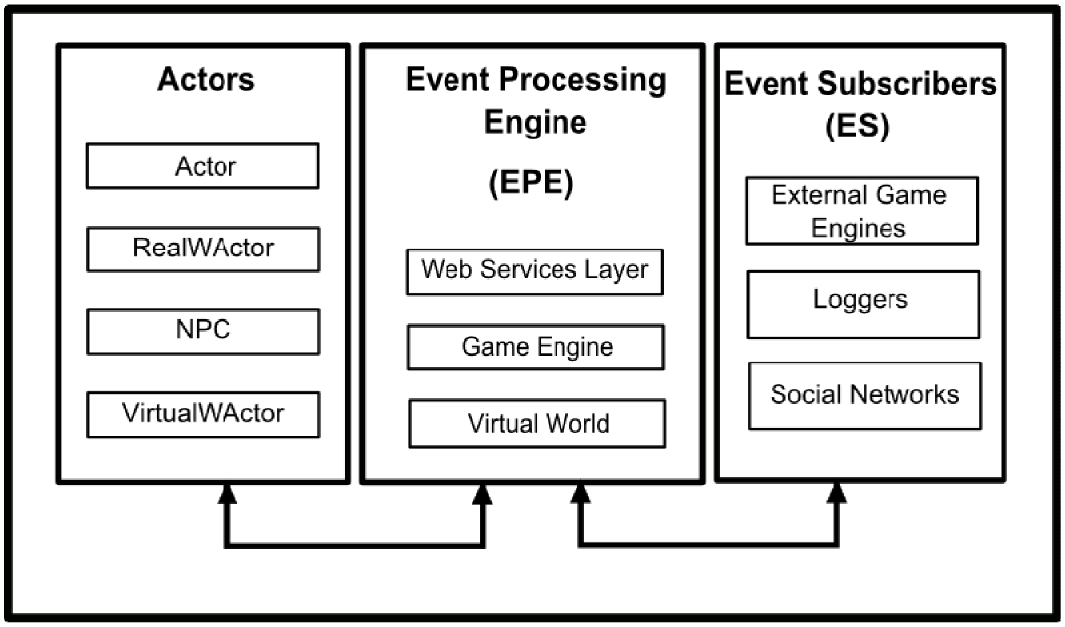

Figure 1. Constituent Components of the fAARS Platform.

As shown in Figure 1, the fAARS platform consists of three run-time components. From left to right, these components are: a) the Actors through which the platform senses its environment in the real and the virtual worlds and the devices through which they interact with the game play, b) the Event Processing Engine (EPE), and c) the Event Subscribers (ESs). In this section, we explain every component in terms of its design and interaction with other components and its implementation.

Let us assume that we want to design and implement fAARS-Pacman in the *Astral Projection* game-play modality using the fAARS platform. In this modality, Pacman will be walking around in a specific physical area decoding QR codes that represent the “pills”. At the same time, Pacman will be chased by the Ghosts in a parallel virtual world. The location of Pacman in the virtual world will be updated based on the GPS and QR codes data, translated to locations in the virtual world. In order to run this game in fAARS, we need of the following constituent components as shown in Figure 1 from left to right:

1. ***Actors***. In the context of the fAARS platform, every real and virtual player is represented by an instance of an *Actor* component in the game, which enables the player to interact (and detect interactions) with players across the real and virtual worlds. These interactions are *Events*. In the case of fAARS-Pacman, Pacman and the Ghosts are the game *Actors*.

2. ***Event Processing Engine***. This is the component that receives the *Events* produced by the *Actors* in order to process the game rules and update the state of the *Actors* accordingly. Its functionality is exposed through web services, and in terms of implementation, it is partly supported by the Virtual World component. The ***Virtual World,*** a part of the Event Processing Engine, is responsible for informing the Event Processing Engine about virtual *Events* and updating the virtual location of Pacman during game play.

3. ***Event Subscribers.*** The fAARS platform provides a web service that allows external systems to register as observers of a game in fAARS. This is to allow external components to provide further downstream functionality to fAARS games as if they were part of the platform. In the case of fAARS-Pacman, this can be used to attach a logger that will record every movement of the *Actors* in the game for mobility-study purposes.

The fAARS platform has been designed and implemented according to the Event-Driven SOA (Service Oriented Architecture) architectural style [39]. The fAARS components interact with each other through a well-defined RESTful interface that allows for the extension or replacement of already implemented parts.

3.4.1 The Actors

In order to start creating fAARS-Pacman and any MAARG, the game rules have to be defined and entered into the database. But before creating the game rules, the *Actors* of the game have to be identified. In the context of the fAARS platform, an *Actor* is a representation of a player in the game. In the case of fAARS-Pacman for example, there are five *Actors*, one Pacman and four Ghosts, and each *Actor* has one game role. The fAARS platform supports three different types of *Actors*, each with different capabilities in the game. As shown in the class diagram of Figure 2, an ***Actor*** can be either a real world player (***RealWActor***), a virtual world player (***VirtualWActor***), or a ***NPC*** which can only exists in a virtual world.

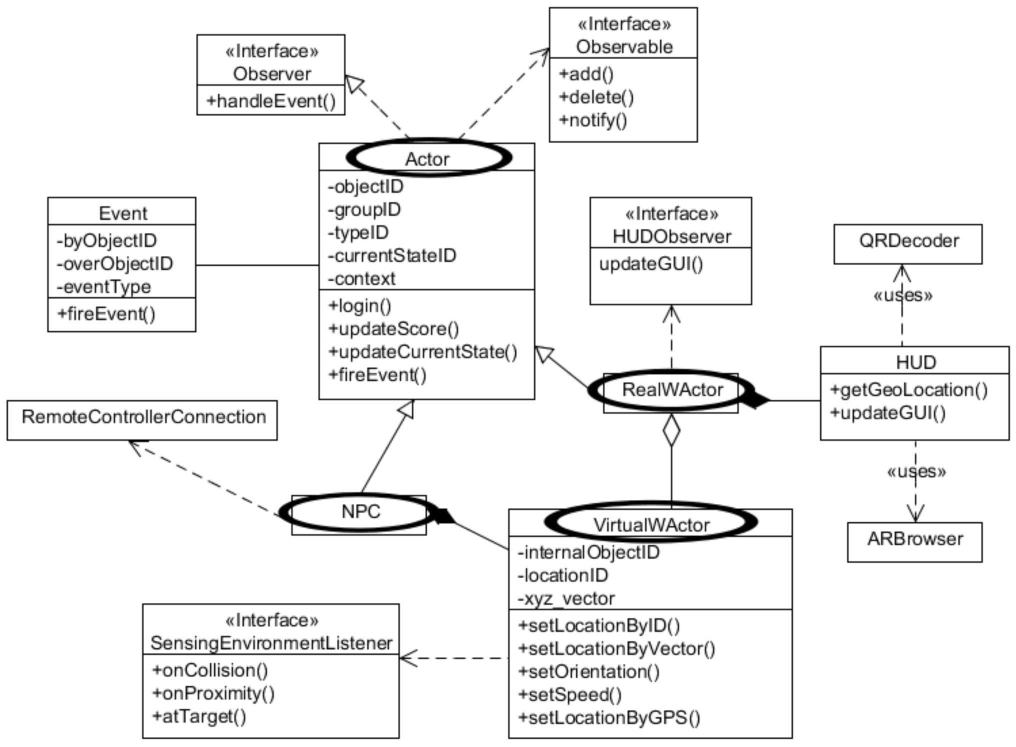

Figure 2. The Logical Design of the set of classes responsible for the Actors

As depicted in Figure 2, ***Actor*** is the parent class of the three type of *Actors* that are supported in fAARS, and it is extended by ***RealWActor***, and ***NPC*** which both contain a ***VirtualWActor***. These *Actors* generate *Events* during game play. An *Event* describes an interaction between two *Actors,* which are the generator and the recipient of an *Event*. In the context of fAARS, an *Event* consists of a) an event *generator*, b) an event *type*, and c) an event *recipient*. Given a new *Event*, the Event Processing Engine proceeds to a) update the state of players in the game, and b) progress the game to reach any specific goal(s).

For instance, in the case of fAARS-Pacman, the Pacman and the Ghosts can sense and generate *Events* during game play, such as “Pacman eats a Ghost”, “a Ghost eats Pacman”, or “Pacman eats a powerpill”. In each of these *Events*, either Pacman or the Ghost is an *Event* generator, and in the context of fAARS, any game character and object that senses and generates *Events* is an *Actor*. In order to further explain what type of *Events* these *Actors* can sense and generate, a description of each type of *Actor* depicted in the diagram of Figure 2 follows.

Actor

As shown in Figure 2, every player in fAARS is an *Actor*. An *Actor* contains information that identifies it as a unique game character or object. For example, in fAARS-Pacman, Pacman and the Ghosts are the main game characters in the game. An instance of *Actor* for each of these characters is created in order to identify each player during game play. The attributes of *Actor* support the identification and grouping of game characters. A brief description of each attribute of *Actor* is provided in Table 2.

Table 2. Set of Attributes of an Actor

Atribute	**Description**	**fAARS-Pacman Example Value**
objectID	Unique identifier of a game character or object in a game.	blueGhost1, blueGhost2, redGhost1, redGhost2, Pacman
groupID	Unique identifier of a group of game characters or objects.	blueGhosts, redGhosts
typeID	Unique identifier of one type of game character or object. This is the parameter where the game role can be specified.	ghost, pacman
currentStateID	Unique identifier that stores the current state of the game character or object.	ghost_running, ghost_chasing, pacman_running, pacman_chasing
context	A variable that stores the active world scenario of an *Actor*. This is updated every time a player logs in to a game.	virtual_world, real_world
score	Points generated during the game.	300, -200

As shown in Figure 2, every *Actor* instance can register as an observer of other *Actor* instances. This is an approach known as the Observer Pattern. This is very useful for game creators to define rules where some *Actors* are interested in being notified about *Events* of other *Actors*. For example, in fAARS-Pacman, when Pacman eats a “powerpill”, all four Ghost are notified about this particular *Event*, hence the current state of all the *Actors* in the game is updated, and the *HUDs* of the *Actors* receive a notification about this state change. This allows the *Actors* in the game to act accordingly, in this case, to run away from Pacman.

In any fAARS game, generic *Actor* objects can be instantiated to represent game items or objects, but nothing interesting can happen in a game of only *Actor* instances. As shown in Figure 2, an *Actor* instance cannot generate any type of *Events*. In order to create interesting fAARS games, either in the real or a virtual world, objects of type *RealWActor, VirtualWActor* or *NPC* have to be instantiated.

RealWActor

In fAARS, every player is represented in the game with an object of type *RealWActor*. For instance, in fAARS-Pacman five players participate, one acting as Pacman, and four more acting as the Ghosts. Each of these participants is represented in the game by a *RealWActor* instance. These types of *Actors* contain a *HUD* object, through which players are able to log in to the game, receive game state changes, and interact with other *Actors*. Features like decoding QR codes, displaying augmented reality content on the physical world, and updating the location of a player using the GPS are also responsibilities of a *HUD* object. This has been implemented as a cross-platform native mobile application that senses the real world environment through QR decoding, augmented reality content, and GPS. Table 3 provides a description of real world *Events* that a *RealWActor* can generate in a fAARS game through the *HUD* object.

According to the diagram of Figure 2, a *RealWActor* also contains a *VirtualWActor* object. The major responsibility of a *VirtualWActor* object is to bridge the real and virtual worlds, when a player logs in the game in the real world. In this manner, the game is extended beyond the real world in terms of time and space, allowing real-world *Actors* to interact with virtual-world *Actors*. This is exactly what qualifies as the *Astral Projection* game-play modality, which is partly implemented using Open Simulator. The interactions and communications among *Actors* in the virtual world are supported by built-in components implemented in Linden scripting language[22].

Table 3. Set of Events that a RealWActor Instance Can Generate

Event Type	**Description**	**fAARS-Pacman Example Event**
decodeQR	An *Actor* decodes a QR code.	"Pacman eats a pill", "Pacman eats a powerpill"
updateGeoLocation	An *Actor* detects a change of location based on GPS coordinates.	"Pacman moves", "A Ghost moves"
touchObject	An *Actor* clicks on an augmented reality game character or object using the built-in augmented reality browser of fAARS.	"Pacman eats a pellet", "Pacman eats a powerpill"

NPC

In fAARS, a virtual player can be an *NPC*. It shares similar characteristics and behaviors with a *RealWActor* object, with three main differences.

1. ***The space where the interaction occurs:*** An *NPC* instance can exist only in a virtual world, where a *RealWActor* instance can exist in both worlds. The strong composition relationship of a *NPC* with a *VirtualWActor* forces this type of *Actor* to only exist in a virtual world.

2. ***Who controls these objects***: An *NPC* instance can be controlled by an external system but a *RealWActor* instance is always controlled by a human either in the real or a virtual world.

[22] Linden scripting language http://en.wikipedia.org/wiki/Linden_Scripting_Language

3. ***The type of events they can sense and generate:*** An *NPC* instance can only sense and generate events that happen in a virtual world but a *RealWActor* instance can sense events in both worlds.

VirtualWActor

A *VirtualWActor* object bridges the real and virtual worlds. *RealWActor* and *NPC* instances can interact with each other in the virtual world through *VirtualWActors* instances during game play. This type of objects can sense and generate virtual world *Events*. As shown in the diagram of Figure 2, a *VirtualWActor* instance can sense and generate three types of *Events*: 1) onCollision, 2) onProximity, and 3) atTarget. When an instance of a *VirtualWActor* senses a virtual world *Event*, all the *Actors* across multiple worlds are notified. This set of virtual *Events* is further described in Table 4.

Table 4. Set of Events that a VirtualWActor Instance Can Generate

Event Type	**Description**	**fAARS-Pacman Example Event**
onCollision	A virtual world *Actor* collides with another virtual world *Actor*.	"Pacman captures a Ghost", "A Ghost captures Pacman"
onProximity	A virtual world *Actor* passes by another virtual world Actor at a certain distance or radius.	"A Ghost starts to follow Pacman" Detection of when a Ghost is close enough to Pacman
atTarget	A virtual world *Actor* arrives to a specific spot in the virtual world.	"A Ghost goes to the recovery room after Pacman has captured it" Detection of when a Ghost is at a specific spot

One of the responsibilities of a *VirtualWActor* is to keep track of the location of players in the virtual world. This is essential for games that use the *Mirror* and *Astral Projection* game-play modalities. As depicted in the class diagram of Figure 2, a *VirtualWActor* instance keeps track of the locations of virtual *Actors* in two formats: a) by locationID or zoneID, and b) by a xyz-coordinates vector. Location of *Actors* by locationID or zoneID is vey useful especially in cases when there is not a spatial correspondence between the real and virtual worlds, although this technique can be also used in virtual and real spaces with similar space configurations. This particular format is mostly used when the communication and location detection of a player is done in a very granular way at different points in time. In game-play modalities that use this technique, like *Astral Projection*, the Game Engine is only interested in the last location visited by a virtual *Actor*. Every locationID corresponds to a specific coordinate in the virtual world. Theoretically, this format would also allow to add 1D and 2D virtual worlds, since every physical location would have corresponding mapping zones in a 1D or 2D space, but we have not been able to test this in a game. In other cases, the xyz_vector format is more useful when games have been setup to be played in the *Mirror* game-play modalities. The exact location of a real world *Actor* is translated to the corresponding location in the virtual world, by translating the GPS coordinates obtained from the smart-phone to the corresponding virtual-world coordinates. This is a format that enables the system to keep the real-world player constantly synchronized with its reflection in the virtual world, and it is also related to the type of device that detects and communicates the location of the user, being the most common the smart-phones equipped with GPS. In either format, it is the responsibility of a *VirtualWActor* instance to translate locationIDs and GPS data to the corresponding location in the virtual world.

Summarizing so far, the fAARS platform supports three different types of *Actors* that allow the deployment of MAARGs in different game-play modalities. For instance, fAARS-Pacman has a total of five *Actors*, one Pacman and four Ghosts. Since we want this game to be played in the *Astral-Projection* modality, we will need the deployment of one *RealWActor Actor* representing the Pacman player, and four *VirtualWActor Actors* representing the Ghosts. In this way, the Pacman player will be able to log in to the game using the smart-phone in the real world, and four Ghost players will be able to log in to the game using the smart-phone simulacrum in the virtual world. This will allow Pacman to collect “pills” using the QR decoder in the real world while being chased by the Ghosts in the virtual world. While the Pacman *Actor* moves around in the real world, the smart-phone will be continuously sending the GPS coordinates to its corresponding

VirtualWActor instance running in the virtual world in order to update the virtual location of Pacman.

3.4.2 Defining the ECA Rules of MAARGs in fAARS

Although a game-authoring environment has not yet been implemented as part of the fAARS platform, the process of entering the rules of a game into the fAARS database is rather straightforward. The first step is to identify the *Actors* of the game. For instance, there are five *Actors* in fAARS-Pacman, Pacman and the four Ghosts. Since this game will be played in the *Astral Projection* modality, we need the deployment of one *RealWActor* instance that will play the role of Pacman, and four *VirtualWActor* instances that will play the role of the Ghosts. Based on this information, we can proceed to define the rules of fAARS-Pacman.

The regular arcade version game of Pacman has multiple rules that can be represented as *Event-Condition-Action* rules that can be understood by fAARS. In order to illustrate the process, we will use two Pacman rules as examples:

1. A Ghost can capture Pacman while Pacman has not eaten a “powerpill”
2. Pacman can capture Ghosts after Pacman eats a “powerpill”.

Three steps are necessary to define these Pacman rules as fAARS *ECA* rules:

First, an *Event* must be defined to cause the first *ECA* rule to fire. This implies that we need to translate the actions of an *Event* generator over an *Event* recipient to the corresponding *Event* type in the real or the virtual world. The “capture” action is translated to “collision” events, as follows:

1. The “A Ghost(generator) *captures* Pacman(recipient)” action is translated to “A Ghost *collides* with Pacman”.
2. The “Pacman(generator) *captures* a Ghost(recipient)” action is translated to “Pacman *collides* with A Ghost”.

These two ECA rules reference the *onCollision Event* that can only happen in a virtual world through *VirtualWActor* instances. Moreover, a third *Event* can be inferred from the two Pacman rules above. This is when Pacman eats a "powerpill". In the context of fAARS-Pacman, this is an *Event* that happens in the real world by decoding a QR code using a smart-phone:

3. The "Pacman(generator) eats a *powerpill*(recipient)" action is translated to "Pacman *decodes* a *powerpill*".

Although a "powerpill" is referred to as the *Event* recipient in this case, it is just an item or game object in the game that allows Pacman to change its state and increase its score. The way to represent this game object in the context of fAARS-Pacman is to create an instance of an *Actor* object (see Figure 2) as the "powerpill".

The next step in the rule-specification process involves the specification of the *Condition(s)* to activate one *ECA* rule. A rule is activated based on the current state of the *Actors* that are involved in an *Event,* the generator and the recipient. A set of states for each of the *Actors* in a MAARG has to be defined based on their game roles. For example, although in fAARS-Pacman there are five *Actors*, there are only two game roles: an *Actor* can be either Pacman or a Ghost. Pacman can be in one of three states at a time: (a) "chasing" a Ghost after having eaten a "powerpill", (b) "running_away" from a Ghost, or (c) "captured". In the same way, a Ghost can be in either one of three states at any time: (a) "chasing", (b) "running_away", or (c) "captured". Using this information, the conditions of each of the *ECA* rules that correspond to each of the *Events* defined in the previous step are:

1. "When Ghost is in the *chasing* state" and "When Pacman is in the *running_away* state".

2. "When Pacman is in the *chasing* state" and "When Ghost is in the *running_away* state".

3. "When Pacman is in the *running_away* state".

Finally, the third step of the process involves defining the set of *Actions* that have to be performed by the platform when an *ECA* rule fires. There are three different types of *Actions* that can be performed by the fAARS platform, which are: (1) update the current state of the generator and/or recipient involved in an *Event*, or/and (2) send a pushed notification to the generator and/or recipient of an *Event*, or/and (3) update the score of the generator and/or recipient of an *Event*. In fAARS-Pacman, for each of the *Event-Condition* values previously defined as part of the *ECA* rules, the set of *Actions* to be performed by the fAARS platform are (1), (2), and (3) for both, the generator and the recipient of the *Events* for all three *ECA* rules. Details about the *Actions* performed by the platform as part of each of the *ECA* rules follows.

ECA rule one: “A Ghost(generator) *captures* Pacman(recipient)”

> *Event:* “A Ghost *collides* with Pacman”
>
> *Condition:* “When Ghost is in the *chasing* state" and "When Pacman is in the *running_away* state”
>
> *Action:* (1) Change the current state of Pacman to “captured”, (2) send a notification to the generator and recipient of the *Event* with information about what just happened in the game, and (3) update the score of the generator and recipient of the *Event* accordingly.

ECA rule two: The “Pacman(generator) *captures* a Ghost(recipient)”

> *Event:* “Pacman *collides* with A Ghost”
>
> *Condition:* “When Pacman is in the *chasing* state” and “When Ghost is in the *running_away* state”
>
> *Action:* (1) Change the current state of the corresponding Ghost to “captured”, (2) send a notification to the generator and recipient of the *Event* with information about what just happened in the game, and (3) update the score of the generator and recipient of the *Event* accordingly.

ECA rule three: The “Pacman(generator) eats a *powerpill*(recipient)”

Event: "Pacman *decodes* a *powerpill*"

Condition: "When Pacman is in the *running_away* state"

Action: (1) Change the current state of all the Ghosts to "running_away", and change the current state of Pacman to "chasing", (2) send a notification to the generator and recipient of the *Event* with information about what just happened in the game, and (3) update the score of the generator *Event* accordingly.

The process of entering the *ECA* rules into the database is also relatively straightforward. There are four tables in the fAARS database that together contain all the *ECA* rules of all the games that run in the platform. One of these tables links a MAARG with all its *ECA* rules. This table is called "ECA_Rules", and it contains as many entries as the number of *ECA* rules created for all the games in the platform. The relationship between one MAARG and one *ECA* rule generates a key value that is exported to three other tables that specify the *Event*, *Condition*, and *Actions* of the *ECA* rule. The table that contains the *Events* entries is composed by three columns, where the generator, recipient, and the type of *Event* can be entered for each of the ECA rules. The *Condition* table is composed by three columns as well, and these columns contain the conditions that the generator and recipient must comply with. In this table we can save the name of a condition, define to whom the condition applies, and the value of that condition. Each row in this table contains as many conditions as are applicable for a particular ECA rule for both, the generator and recipient of an *Event*. Finally, the *Actions* table contains the actions that the fAARS platform has to perform. It is composed by three columns, and contains the name of the action, the recipient of the action, which could be the generator or recipient of an *Event*, and the value that will be used to perform the action. At run time, when the Event Processing Engine receives an *Event* during game play, it extracts from the fAARS database all the key values contained in the "ECA_Rules" table based on the ID of the game, and filters the generated list by querying the *Event* and *Condition* tables in order to compute the corresponding *Actions*. This and other functions are performed by the

Event Processing Engine, which is described in the following section.

3.4.3 The Event Processing Engine (EPE)

The Event Processing Engine is responsible of a) receiving *Events*, b) processing *ECA* rules, c) maintaining the game state, and d) implementing a layer of APIs for the efficient communication among the internal and external components of the platform. Figure 3 shows the internal components of the Event Processing Engine, i.e., a) a Game Engine; b) a Persistence Layer, and c) a Web Services Layer. A description of each of these components follows.

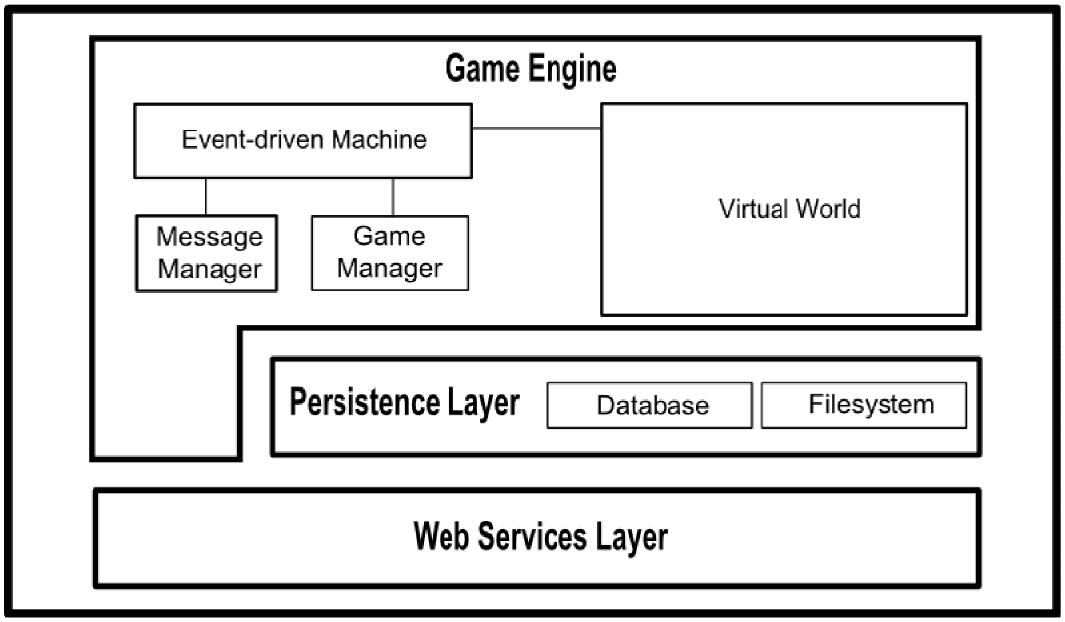

Figure 3. Internal components of the Event Processing Engine

The Game Engine

The game engine is the most important component of the fAARS platform. Its responsibility is to process the game rules during game play and to connect the various internal pieces of the platform to advance the game and update the state of *Actors* in the real and virtual worlds. Figure 4 displays a sequence diagram of how the internal pieces of the game engine interact with each other when a new *Event* is generated by an *Actor*.

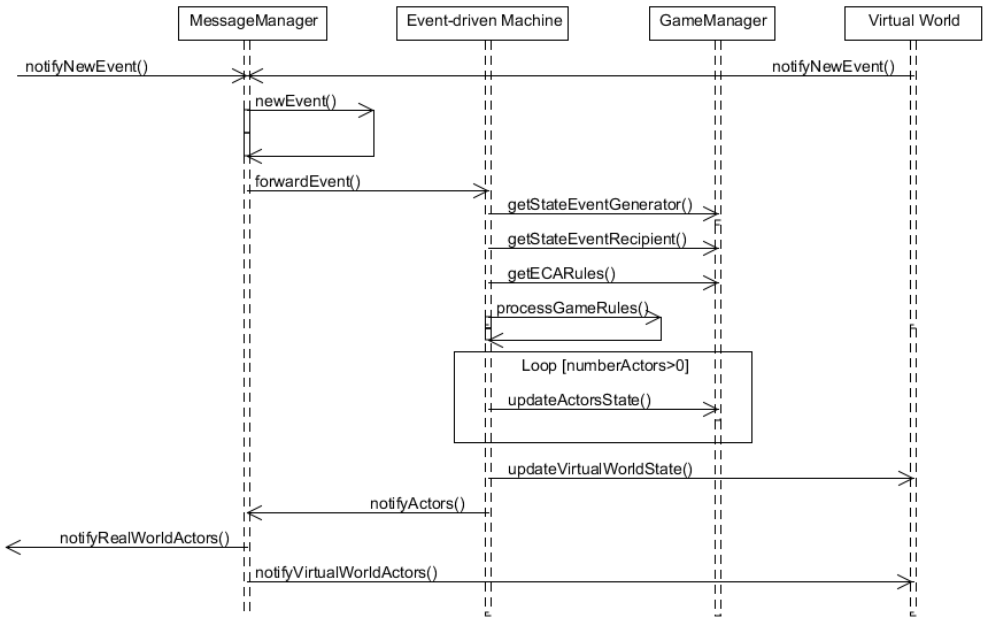

Figure 4. Sequence Diagram of the Internal Components Interaction of the EPE

As shown in Figure 4, every new *Event*, generated by an *Actor* in the real or virtual world, is received by the *MessageManager*. The *MessageManager* is in charge of sending and receiving messages in the Game Engine. Whenever a new *Event* is received, it is always forwarded to the *Event-driven Machine*. The *Event-driven Machine* communicates with the *GameManager,* which queries the fAARS database in order to extract the matching *ECA* rules of the game based on the *Event* and *Condition* values. Based on the matching *ECA* rules, the *Event-driven Machine* performs the corresponding *Actions*, and the state of all the *Actors* in the game is updated and saved in the fAARS database via the *GameManager*. Finally, in order to externally reflect this update, *Actors* in the real and virtual worlds are notified about game state changes via the *MessageManager*. Internally, each of the components of the Game Engine depicted in Figure 4 behaves as an observer of the *Event-driven Machine*. They are responsible for reflecting a game state change in the real and the virtual worlds whenever applicable.

- **The Event-driven Machine**

The main responsibility of the *Event-driven Machine* is to process the *ECA* rules of the games developed in fAARS. It goes through a number of different states in order to process the *ECA* rules with every new incoming *Event*. The internal transition from one state to another is triggered by internal messages, which are recursively fed into the *Event-driven Machine*. Each of these states encompasses a set of activities that allow *Actors* to progress according to the game rules At the implementation level, the *Event-driven Machine* is partly supported by the *Virtual World* component when peer-to-peer interactions are needed in a game in fAARS.

- **The Virtual World**

Every game developed in fAARS can have a *Virtual World* running in parallel during a game. This piece of software acts as the alternate reality component of games developed in fAARS. It is implemented using the commodity virtual world engine Open Simulator, which contains built-in functionality that enables the deployment of fAARS games in a virtual world without having to write any extra code. Extension of games to a virtual world can be defined in terms of space and time. When games are extended in terms of space, it refers to the spatial correspondence between the two worlds. When games are extended in terms of time, it means that time can run slower, faster, or similar in the virtual world in comparison to the real world. This parameter is controlled via the speed at which virtual world *Actors* move, and can be controlled externally by the *Event-driven Machine*. This parameter can be configured in the fAARS database or directly inside the virtual world.

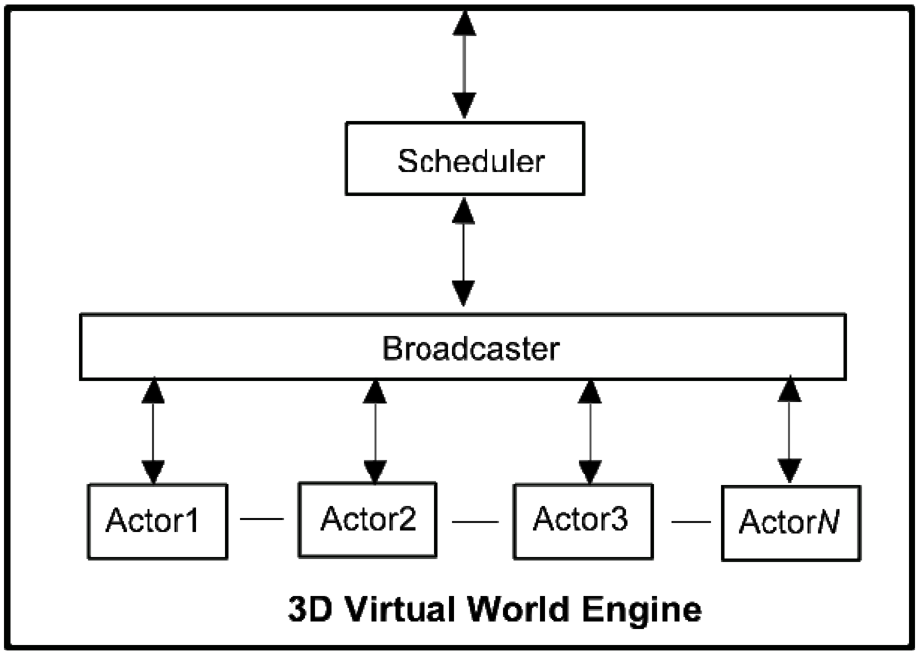

Figure 5. Internal Components of a Virtual World

As depicted in Figure 5, a typical *Virtual World* component consists of:

1. *3D Virtual World Engine:* It is the 3D virtual environment to which games in fAARS can be extended. Generally, the *Virtual World* component setup is done by hand using virtual world tools provided by the Open Simulator package along with the pre-implemented set of virtual objects to create the *Actors* of the game.
2. *A Scheduler:* It is an observer of the *Event-driven Machine*. Its main responsibility is to open an XML-RPC port and forward packets in and out in the virtual world.
3. *A Broadcaster:* It is an observer of all the *Actors* in the virtual world. It translates data and commands between the *Scheduler* and the *Actors*. It also allows *Actors* to communicate with each other, and keeps an updated list of active *Actors* during game play.
4. *Any number of Virtual Actors:* A *Virtual Actor* represents a game item or a player in the context of fAARS. Within the context of the virtual world, a game item or a player corresponds to an *NPC* or a *VirtualWActor* as seen in

the diagram depicted in Figure 2. Virtual Actors are capable of sensing its environment to generate *Events* in real time, and has the ability to move around to any location in the virtual world by itself thanks to a way-finding algorithm pre-packaged as part of fAARS.

It is important to mention that the integration of a *Virtual World* Engine as part of the components of the fAARS platform, specifically Open Simulator, did not have any impact to the design and implementation of the other components of the platform. Open Simulator was one of the last pieces in the platform that was added during the implementation phase.

In fAARS-Pacman, there are five virtual world *Actors*, where four of them are the Ghosts and one of them is the Pacman. They are capable of sensing their environment to generate game *Events*. Whenever a Ghost captures Pacman or the opposite, the *Broadcaster* forwards this *Event* to the *Scheduler*. The *Scheduler* notifies the *Event-driven Machine* about a new *Event*, where the state of the virtual *Actors* is updated based on the *ECA* rules computed.

- **The Game Manager**

The *Game Manager* bridges the Persistence Layer and the Game Engine. It allows the *Event-driven Machine* to internally query the fAARS repository for resources that are used during game play in a fAARS game. These resources include information about the game, the state of *Actors* in the game, and the *ECA* rules for processing by the *Event-driven Machine*. These resources also include the media content used in the *HUD* through which players receive updates about game state changes. The *Game Manager* also manages *Actors* sessions, and keeps track of active *Actors* and their *Actions*. From a high-level perspective, the *Game Manager* is in charge of moving data in and out of the Persistence Layer and keeping an updated list of active *Actors* and their state in the game.

- **The Message Manager**

The *Message Manager* keeps the real and the virtual world *Actors* updated as the game and *Actors* in the game transition from one state to another. Its main responsibility is to send and receive messages in the Game Engine. Communication with the real and virtual *Actors* of a game can happen at any point in time during game play as the *Event-driven Machine* demands it. This is done using the Long Polling protocol that was implemented using a Javascript framework. In the context of a client-server communication, the Long Polling protocol is a communication method where the server holds a request for as long as there is no information available for the client. When the request times out, the client automatically sends a new request to the server. If information for the client is available, the client receives such information and makes a new request every time. This approach was implemented using the Moo-tools[23] javascript framework.

The Persistence Layer

The Persistence Layer is the data warehouse of the fAARS platform. As shown in Figure 3, it is composed by 1) a database and 2) a file system. These two pieces are internally connected to provide all the necessary data store needs to any game developed in fAARS. The database stores most of the game information. It is a reliable and secure data store that internally exposes a RESTful API to other components in the platform in order to perform Create, Read, Update and Delete (CRUD) operations. This API uses a UNIX socket connection to securely perform the CRUD operations.

The database contains six different types of data: (a) information about users and their devices, where the status of the player in the game is saved such as score, number of challenges completed, and current game level; (b) logged traces, as a sequence of locations visited during game play, as triples of QR code ID or GPS coordinates, user ID, and timestamp; (c) virtual-world metadata, including information such as the URL, channel, port, and size of the virtual world; (d) augmented-reality metadata, including information such as the GPS coordinates where the browser is able to display content, and the URL base to download the

[23] Javascript Framework Moo-tools http://mootools.net/

augmented reality content from; (e) game metadata, i.e., the components that are used by a game such as *Virtual Worlds* or augmented reality content, time of deployment and game name; and finally, (f) the *ECA* rules of every game in fAARS. The database was implemented using the MySQL database manager, which uses the MyISAM data store engine for better performance.

The file system works in collaboration with the database manager. It stores and retrieves media content files that are used by a game. These files are referenced from the database using URLs, which point to the fAARS repository server and the filename. The file system is accessible through a RESTful interface that runs behind an Apache Web server to allow storing and retrieving of files easily. This flexible interface enables the *HUDs* in the real and virtual worlds to download all the media content of a game.

The Web Service Layer

The functionality of every internal component of the Event Processing Engine has been exposed as a set of RESTful APIs, to allow external and internal components of the platform to communicate with each other in a flexible manner. Moreover, this approach allows the internal and external pieces of the Event Processing Engine to be upgraded and modified without any impact to other pieces. It also allows scalability at different levels, because we can migrate individual components to specialized servers for a better performance. The Web Service Layer supports games to be deployed almost anywhere with Internet access, and players at different locations can play fAARS games as long as they have an Internet connection available.

3.4.4 The Event Subscribers (ESs)

The Event Subscribers are external systems interested in being notified about game *Events* in fAARS. These external components provide further downstream functionality without directly affecting the internal functionality of the platform. The fAARS platform exposes a web service through which external systems can register as observers of games in fAARS.

Table 5 provides a sample list of external systems that can be added to the platform. As shown in Table 5, external systems can be classified as Active or Passive in the context of fAARS. Active systems can push events to the Event Processing Engine, and passive systems are only interested in being notified about new *Events* without performing any further actions. In either case, as long as the external component is capable of handling the Long Polling messaging protocol and understanding JSON encoded messages, they can register as observers to receive messages that contain: a) information about the *Event* generator, b) information about the *Event* generated, and c) information about the recipient of the *Event*.

Table 5. Sample list of external systems that can be attached to the fAARS platform.

Name	***Type***	***Description***
External Game Engines	Active	Systems that can extend the rules of a game. These type of systems push *Events* to the fAARs platform based on their own set of rules.
Loggers	Passive	Systems that log *Events* of games in fAARS.
Social Networks	Passive	Social networks like Facebook and Twitter can be attached in order to publish interesting *Events* that can happen in a fAARS game.

In the case of fAARS-Pacman, in order to study the mobility data generated by the *Actors* moving around in the real and virtual spaces, a Logger can be attached to save the players' location information in real time as the game happens. Furthermore, as seen in Table 5, in order to make the fAARS-Pacman more interesting to external viewers, we can create a simple script that can register as an observer of the game in order to publish to Twitter or Facebook the most interesting *Events* that happen during the game. This can attract other potential players to watch games asynchronously from anywhere in the world.

Chapter 4: Our Experience with the fAARS Platform

We have developed two games using the fAARS platform. These games were used in two experiments performed in collaboration with two Departments of the University of Alberta. The first game is titled “Human Geometric Orientation”, and it was developed in collaboration with the Psychology Department to test subjects on their performance in geometric orientation. The second game is called “Outbreak: Safety First”, and it was implemented in collaboration with Dr. Sarah Forgie from the Faculty of Medicine to test the fAARS platform as a potential educational tool.

These two games exploit different capabilities of the platform, and both of them have generated very interested results in their corresponding areas of study where they were used. The deployment of these two games shows the flexibility and robustness of the platform. It is worth mentioning that while the two games were running, we did not experience any technical issues related to its performance. Although the number of simultaneous players was not great, this is still strong evidence for the robustness of the platform. For the deployment of these two games, the platform was installed across three different servers. One server was in charge of running the *Virtual World Engine* component, another one was in charge of processing the game rules, and the third one was the message manager in charge of receiving and sending messages between the players and the fAARS platform. The component that consumed most of the computational resources was the *Virtual World*. In terms of scalability, the *Virtual World* component would be probably the first component that would have to be migrated to a very powerful machine. In our experiments, Open Simulator was deployed on a machine equipped with four cores and 16 GB of RAM memory, in order to be able to handle at least 20 users at once.

In the following sections, a description of the games and their objectives is presented, the run-time components of the platform that were used for their deployment are described, and the results obtained from these two studies are discussed.

4.1 The "Human Geometric Orientation" Game

Almost all kinds of animals, and humans in particular, need to orient in their environment for survival. One common feature used for orientation is the geometry of the space. Due to the complexity of creating the proper environment to test how humans orient themselves, there has been a limited number of studies in this area. Furthermore, the conflicting findings on how humans and animals encode the geometry of the space, in particular the theoretical importance of encoding angles for orientation[40], suggest the need for further research.

In order to test the set of theories around the subject of geometric orientation of humans, in the summer of 2011, 99 University of Alberta undergraduate students were invited to participate in a game implemented on fAARS. In this game, participants were tested in a first-person navigable virtual world, where they were trained to locate two geometrically equivalent corners in a number of parallelogram-shaped rooms. The task could be solved based on identifying three features: angular amplitude, relative wall lengths, or a principal axis. At the beginning of the game, participants did not know of a clear distinction between training and testing. They were instructed to determine which corner was correct, and walk into it by using only the arrow keys of a keyboard. At first, they did not know which corner was correct, so they were told to guess. During the game, correct choices would earn them 5 points, incorrect choices no points, and they had to try to get as high a score as possible. The participants had to choose the correct corner in every room until completion. The objective of the game was to earn as many points as possible by visiting all the rooms while avoiding wrong choices if possible. In order to move from one room to another, a minimal set of correct choices in every room had to be completed. This was done by automatically re-locating the view of the participant in the virtual world right in the middle of the room, and facing a random direction every time. Upon visiting all the rooms in the virtual world, they were placed in a square room with a debrief message displaying their score, their position in the leaderboard, and instructing them to turn off the monitor.

4.1.1 Game Implementation on fAARS

In the context of fAARS, the "Human Geometric Orientation" game was fully developed in a virtual world, and participants played this game in the *Dreaming* game-play modality. All interactions between participants and the corners in the rooms happened entirely in a virtual world, thus only virtual world *Events* were generated during game play. In order to develop this game, the following fAARS components were used:

1. ***Actors.*** A total of 99 *VirtualWActor* instances with unique IDs were created to allow participants to interact with the corners in each of the rooms in the virtual world. Each corner was represented by an *Actor* instance in order to detect the recipient of an *Event*. The *VirtualWActor* instances were in charge of informing the Event Processing Engine of every new virtual *Event* to process the *ECA* rules of this game. There were no game roles in this game.

2. ***Event Processing Engine.*** The Event Processing Engine was responsible for detecting when a *VirtualWActor* chooses the right corner in every room. The *Event* type used to define every *ECA* rule was the *onCollision Event*, and the *Condition* for each of these rules to be activated was that the generator and recipient of an *Event* has to be included as part of an *ECA* rule without considering their current state in the game. The *Actions* performed by the fAARS platform when a rule was activated consisted on increasing the score of the *Actor* if a choice was correct.

3. ***Virtual World.*** This was an essential component for this game. It is where all the game play happened in the *Dreaming* modality. The rooms were setup manually in the virtual world, and the corners in each of the rooms were instantiated as *Actors* using invisible virtual objects that were added to each of the corners of the rooms.

4. **Event Subscribers.** There was one Event Subscriber in this game that provided further downstream functionality. It was an Active component, and

was in charge of keeping track of the number of tries of every *VirtualWActor* instance for each of the rooms in the virtual world, and updating the location of the *VirtualWActor* instances based on the number of tries.

4.1.2 Experimental Results

From the set of 99 undergraduate students recruited for this study, only 77 were able to finish the experiments correctly. Twenty-two students were excluded because they did not complete the training phase within 30 minutes or scored lower than the 70 percent of the control population. Details about the experimental results of this experiment have been submitted to the *"Psychonomic Bulleting & Review"* journal [19]. Part of the conclusions presented in the submitted article demonstrate that humans are able to encode and navigate using the angular amplitude feature of the space. Moreover, even when angular information is removed, humans can still orient well by encoding wall length instead.

4.2 The “Outbreak: Safety First” Game

"Serious games" are increasingly recognized as effective learning tool among adult learners - in particular, those who have grown up in the digital age and are comfortable with digital literacy, interactivity and immediacy [41, 42]. Although research in the area is still in its infancy, there is also evidence to suggest that their use in the education of health professionals is valid and worthy of further investigation [43]. For instance, games can be used to educate inter-professional health students about the proper procedures to prevent the spread of infections while taking care of their patients. According to Dr. Sarah Forgie, in healthcare, low compliance rates with infection-prevention procedures, such as hand hygiene and use of personal protective equipment (PPE), are believed to be responsible for a large proportion of nosocomial infections [44, 45]. There are approximately 220,000 cases of nosocomial infections in Canada and they lead to 8,000 deaths per year [46]. The corresponding figures in the USA are two million cases per year leading to roughly one hundred thousand deaths, costing more than thirty billion dollars (US) [47]. Educational interventions such as poster campaigns, lectures and contests to promote compliance are abundant, but they rarely have long-term effects [48, 49].

We hypothesize that an educational intervention in a virtual world and mobile game formats will not only educate users about the proper methods to prevent the spread of infection, but also reinforce positive habits and increase compliance with hand hygiene and use of PPE. In the context of the HLTHSIM project in the GRAND[24] Network, and in collaboration with Dr. Sarah Forgie of the Faculty of Medicine, we have built a mobile game that can help health professionals to learn about the basic protocols involved in infection prevention and control – both for the healthcare worker and the patient. As part of our study on using games to train health professionals, we envision that one the potential impacts includes changes in training and management of all healthcare workers, including nurses and physicians. It may also change the way hospitals and other work places are designed, so as to maximize the opportunities for hand hygiene and use of PPE. If compliance increases, there will potentially be fewer nosocomial infections.

One of the objectives of the fAARS platform is to support the development of "serious games" through which players can learn about spaces and places and the activities that occur within them, in a fun and motivating way. We have used fAARS to develop such a mobile game, "Outbreak: Safety First", through which students in health sciences may learn about the basic protocols for infection prevention and control that they will have to follow as professionals working in a hospital. The "Outbreak: Safety First" players are medical students, nurses and physicians. Through their mobile devices they can scan QR codes as they move through the real-world hospital. Some scan events may cause them to become infected. The cycle of their infection and its contagion to other areas, patients and players is controlled through a simulation, configured with simple infection transmission rules. The activities of the players, the simulated viruses, and their

[24] Graphics, Animation, and New Media http://grand-nce.ca/

hosts are reflected in a virtual world, configured to reflect the layout of the real-world space. As the game evolves, players "get infected" through actions dictated by the game (scanning QR codes attached on infected landmarks) or through the exposure of their representational avatars to virtual viruses in the simulation running in the virtual world. When this happens, they are informed through a message to their mobile device, at which point, they have to choose appropriate measures to protect themselves and the patients.

We have conducted pilot tests of the game prototype with groups of inter-professional health students, to analyze the way that medical students and educators play the game, and to assess its suitability and effectiveness as an educational tool. The findings will contribute to the knowledge of the use of this medium in health professional education, and to the research in the area of innovative educational methods for improving patient safety.

4.2.1 Game Objective and Rules

First-year medical students put their skills at test on how to take care of themselves by wearing the proper equipment when visiting/assessing patients in different given scenarios during the game. The scenarios through which they were trained in this game are as follows:

1. *Routine practices (RP) (also known as Universal Precautions):* This scenario applies when the caregiver does not know anything about the patient and has to assume that there is a potential risk of getting infected with some virus.
2. *Contact (C):* This scenario applies when the caregiver knows that she/he has to avoid contact with an infected patient or object, assuming that there is no potential of a viral infection.
3. *Droplet (D):* This scenario applies when there is a risk of potential viral infection through another patient at a distance of less than one meter.

4. *Droplet+Contact (D+C):* This scenario applies when the caregiver knows there is a potential risk of infection through contact or proximity to an infected patient.
5. *Airborne (A):* This is the most dangerous scenario with two variants. In the Airborne-within-a-room (A1) scenario the caregiver is aware of infections flying around in the room. This is typically the type of scenario where there is a patient inside the room where a very dangerous airborne infection resides. In the Airborne-within-an-area (A2) scenario, there is a potential risk of an airborne infection within a given area, but the caregiver does not know where exactly it is, so he/she have to wear a respirator/mask at all times to keep working.

The objective of this game is to train students through all given scenarios by making them wear the proper equipment in order for them to survive and keep playing the game. The completion of one scenario is worth 100 points, and as bonus points players can increase their score by decoding QR codes that allow them to take challenges in the form of questions. These questions are related to PPE procedures and infections prevention. A correct answer is worth 20 points, and an incorrect one is worth -5 points. At the same time, airborne viruses moving around in a parallel virtual world can also infect the players, and players can infect other players via direct contact or droplet. In order to move, players have to scan QR codes in the hospital, where some of the locations have a bacteria attached to it, which means that if the player scans a QR code that is affixed to an infected location, the player gets infected. If a player gets infected, he/she loses 20 points. The player who completes all the scenarios in the game with the highest score wins.

4.2.2 Game Setup and Deployment

Taking advantage of the capabilities of the fAARS platform, we wanted to evaluate (a) whether practicing one's knowledge in the context of a game such as "Outbreak:Safety First" helps with learning and (b) whether the spatial context of the game (and its verisimilitude to the real environment where the students are expected to practice as professionals) plays a role when it comes to learning.

To that end, we developed two versions in addition to the mobile version of the game. Using the same set of game rules and objectives, the two additional versions differ from the original one in terms of the space in which the game is played, and the way that participants interact with it. The two additional versions were deployed in a virtual world, allowing participants to play this game as a regular desktop video game in two different game-play modalities.

As seen in Figure 6, the second game variant uses the virtual world as a clickable 2D map (virtual 2D version), where participants have a top view of a Pacman board, on which players and viruses interact with each other, and participants can click on the items of the game in order to move around and play the game. This is a very similar approach to the regular Pacman game, where viruses are the Ghosts, and Pacmans are the medical students, all of them playing at the same time. Feedback to players was given in a *HUD* running on a smart-phone simulacrum in the virtual world.

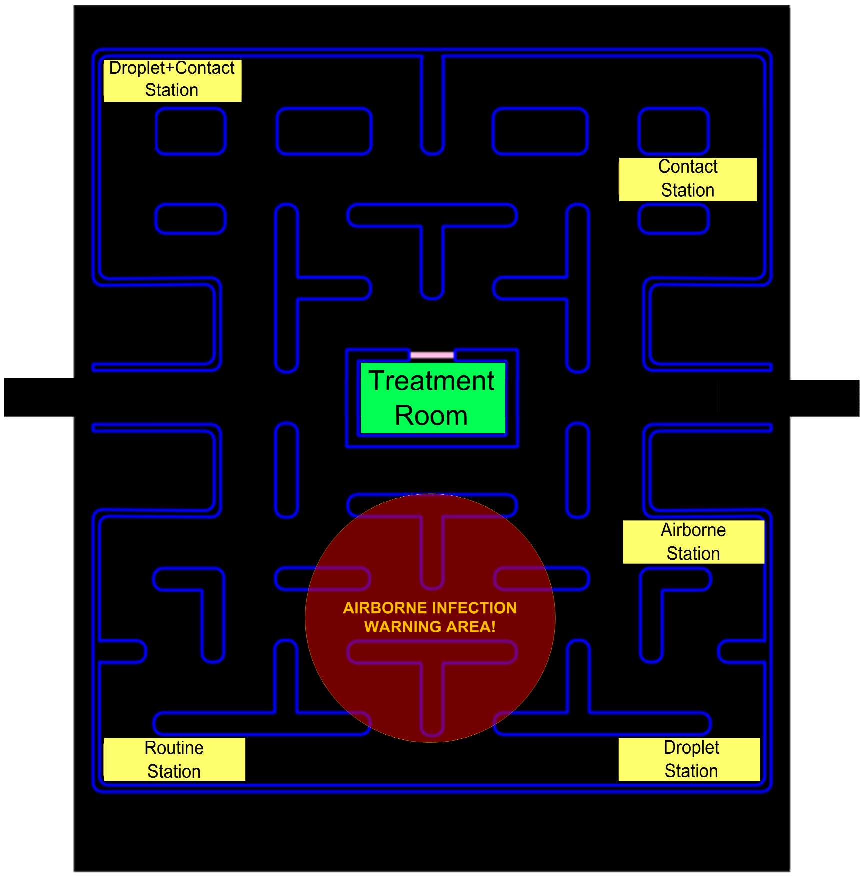

Figure 6. Virtual 2D Version of "Outbreak:Safety First"

As seen in Figure 7, the third game variant takes place in a 3D virtual world, where the virtual space configuration is similar to the physical space where the mobile version takes place, and players have a first-person view in a navigable virtual world. The only difference between the mobile version and this one was that players had to click on the items of the game instead of decoding QR codes, and feedback was given in a *HUD* running on a smart-phone simulacrum in the virtual world.

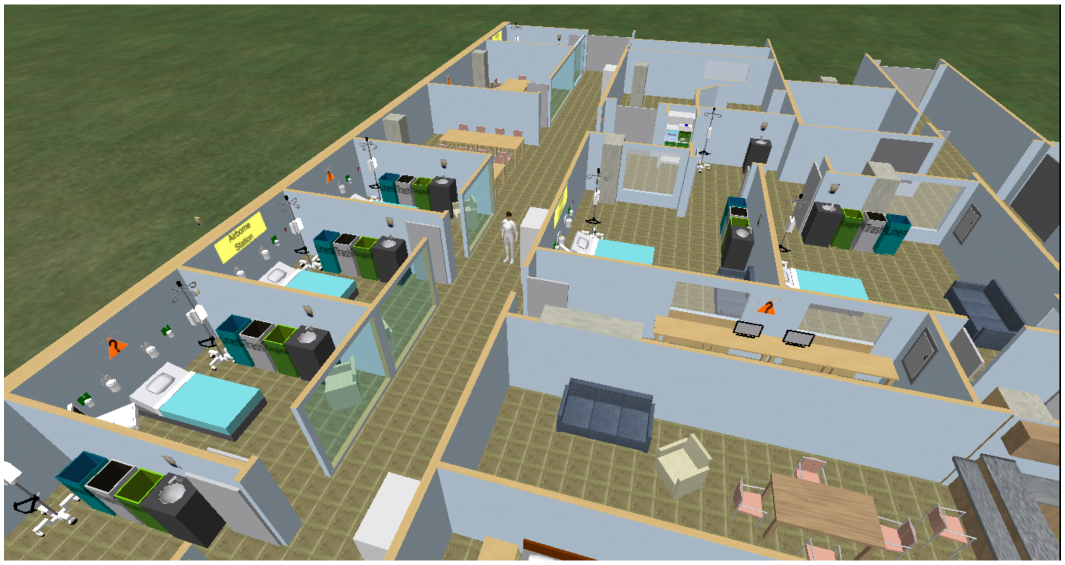

Figure 7. Virtual 3D version of "Outbreak:Safety First"

In all three versions, players are able to play a game with the same objectives and rules, including the style in which feedback was given to players in their *HUDs*. This is a very important aspect of the games, because we wanted experiences across different games to be as similar as possible in order to be able to compare results from our experiments across different versions in terms of the general experiences of the players with the game, and their level of enjoyment and learning.

These three variants of the game were deployed at the Edmonton Clinic Health Academy building (ECHA building). The mobile version was deployed around the "Smart Condo" simulation space of ECHA, where QR codes were added to different locations, and participants had to download a mobile application to their smart-phones in order to play the mobile version of this game. The virtual 2D and 3D versions were deployed over a number of machines in a computer lab in the ECHA building, where an OpenSim client was installed in order to be able to connect to our virtual world servers and the game.

4.2.3 Game Implementation on fAARS

The "Outbreak:Safety First" game extensively exploits the capabilities of the fAARS platform across its three versions. In the context of the fAARS platform, the three versions of this game refer to two different game-play modalities. The mobile version was deployed in the *Astral Projection* modality, where players had a representational avatar walking around in a parallel virtual world with viruses flying around in order to try to infect players during the game. The virtual 2D and 3D versions were deployed in the *Dreaming* modality, where every player had to log in to the game in the virtual world, and interactions among players and viruses was performed in the same virtual environment. Certainly these two game-play modalities exploit different components of the fAARS platform, specifically the type of *Actors* that had to be instantiated in order to allow participants to play this game. In order to develop this game, the following fAARS components were used:

1. ***Actors.*** A number of *VirtualWActor* instances were created for the virtual 2D and 3D versions of the game. This allowed players to log in to the game in the virtual world and interact with all the *Actors* in the game. For the mobile version, a number of *RealWActor* instances with their corresponding *VirtuaWActor* instances were deployed in order to allow physical players to interact with the QR codes in the real world, and to allow them to interact with other *VirtualWActor* instances in the virtual world. Finally, a number of *NPC* instances were deployed to allow viruses in the virtual world to try to infect the *VirtualWActor* instances during game-play.
2. ***Event Processing Engine.*** The Event Processing Engine was responsible for computing the *ECA* rules of the game for all three versions. Based on the *Events* generated by the *VirtualWActor* and the *NPC* instances in the virtual world, and the QR codes that were decoded during the game by the *RealWActor* instances, the corresponding *ECA* rules were processed in order to update the state of all the *Actors* in the game. The *Event* type used to define almost every *ECA* rule of the game was the *onCollision Event*, which was essential to detect when a *VirtualWActor* representing a player was infected by an *NPC* in the virtual 2D and 3D versions of this game. Additionally, in the mobile version of this game, the *decodeQR Event* was used to define a subset

of the *ECA* rules in order to detect when a *RealWActor* decodes a QR code to update its location in the virtual world and advance the game.

3. ***Virtual World.*** This was an essential component for all three versions of this game. The *Astral Projection* and *Dreaming* game play modalities that supported the mobile and virtual 2D and 3D versions of this game use the *Virtual World* component. In the *Astral Projection* modality, the virtual world is used to run a simulation, where *RealWActor* instances could be infected by *NPC* instances that represented viruses flying around in the game. In the *Dreaming* modality, it is in the virtual world where the interactions among all *VirtualWActor* and *NPC* instances happened during game-play.
4. ***Event Subscribers.*** There were two Event Subscribers in this game that provided further downstream functionality. Both of these external components were used in all three versions of the game. The first was a repository of questions, which allowed players to increase their score if they correctly answered a question. The second component was a mobility controller that allowed to move *NPCs* around in the virtual world in a pseudo-randomly fashion. The algorithm used to update the location of the *NPCs* was a very simple one: it pseudo-randomly assigned a different location to an *NPC* every time it arrived at the target previously assigned.

4.2.4 The Experiment Methodology

In order to run our experiments using different versions of "Outbreak:Safety First", recruitment of first year medical students was performed through Dr. Sarah Forgie. She invited a number of her former students to participate in our study that we called *"Assessing the Educational Impact of Course Gamification (in Epidemiology)"*.

There were three phases in the exepriments In the first phase, students took a short test about the use of PPE and general procedures to avoid the spread of infections. Subsequently, they filled out a questionnaire about their experiences with previous educational games and their expectations about our game, and they were asked to describe their feelings about computer-assisted instruction in general using a rating table.

As a second phase, students received instructions on how to play the version of the game to which they were randomly assigned.

In the third phase, students took a follow-up questionnaire including questions about their experience with the game and about their feelings about computer-assisted instruction in general using the same rating table as before, and a second test about PPE and procedures was delivered to the students via email.

Their participation in the experiments took approximately 1.5 hours in total. During the game play of each of these games, the fAARS platform recorded the communications and interactions among the players, including their multiple locations during the game, their performance in game, and the different actions taken by players during game play.

4.2.5 Discussion of Preliminary Experimental Results

During the period of December 14th - 16th of 2011, a total of 36 undergraduate first-year medical students of the University of Alberta played and enjoyed "Outbreak:Safety First" in one of its three variants: virtual 2D, virtual 3D, and mobile versions. At the beginning of each experiment, the participants were asked to describe their expectations for this game in a pre game questionnaire. None of the students knew anything about the game previous to each of the three experiments, not even the version of the game that they were going to play. The only thing that they knew prior to each experiment was that this game would help them to learn more about PPE procedures and how to avoid the spread of infections. According to comments provided in the pre game questionnaires, most of the students reported positive expectations about playing an educational game as part of an experiment. In our collected data, 70%[25] of the participants provided positive comments such as the following:

[25] The classification of the comments in two positive/neutral/negative was done by the author.

"*I'm hoping that it will be fun and an enjoyable way to learn",*

"I think it will be interesting to see what the game is like, and how it might help in learning",

"An interactive opportunity that will give me a chance to practice and use my knowledge, and hopefully gain some too!", and

"To have fun learning".

The rest of the comments, about 30%, were neutral comments such as

"not sure what to expect",

"I'm entering it with a neutral attitude without expectations.", and

"Not sure what to expect", among other neutral comments.

The set of positive comments clearly suggests the willingness of most of the students to participate in games for learning as part of their education. Subsequently, during the game, the participants had very few questions about how to play the game or how to use the system for all three variants, which provides strong evidence that the instructions provided to the participants prior to the game were sufficient. As part of the instructions on how to play the game, we used a couple of video tutorials for the participants to watch, and very few questions were asked afterwards. When all the participants finished playing their assigned version of the game, we showed them the final leaderboard, and some of the participants commented about a few fun aspects that happened during the game. Table 6 presents a set of aggregated stats of all three versions of this game.

Table 6. Aggregated stats of the three versions of "Outbreak:Safety First"

Game Version	**Total number of players**	**Average of total score**	**Average number of questions attempted**	**Average number of questions correctly answered**
Virtual 2D	10	649	12	10
Virtual 3D	17	442	4	3
Mobile	9	440	5	4

As seen in Table 6, 10 participants played the virtual 2D version, 17 participants played the virtual 3D version, and 9 participants played the mobile version of this game.

Regarding the average of total scores shown in Table 6, it is important to highlight that most of the participants of the 2D version of this game obtained the highest scores among all three versions. This particular fact is an interesting one, and it has a direct correlation with the number of questions that participants attempted during the game. According to our collected data, 95% of the participants went through all the scenarios in the game, which implies that scenario completion was not the dominating factor in their final scores. However, as we have already mentioned, one of the objectives of this game was to get as high a score as possible, and this could be achieved by finding and trying to answer as many questions as possible during the game. There were a total of 15 questions related to the spread of infections in each version of this game, and all these questions were randomly added to places easily reachable by the human eye at different certain distances, specifically in the 3D and mobile versions. With respect to the 2D version, players had a top view of the game board, and questions were very straightforward to find. We are confident that the relatively low number of questions attempted by participants in the 3D and mobile versions is not related to the location of the questions, since all of them were added right next to the corridors where the game took place, and the configuration of the space was very simple with four corridors forming a square. This hypothesis is also supported by the fact that the number of questions attempted in the 3D and mobile versions do not represent even a half of the number questions attempted in the 2D version. Therefore, we hypothesize that this substantial difference in the number of the questions attempted is related to the relative complexity of the three versions of the game. More specifically, the complexity of the 2D version is very low compared to the complexity of the 3D and mobile versions. From an educational perspective, we found this particular fact to be a very interesting one, since it suggests that when students play a simple game, it allows them to focus more on the actual content of the game. As we will see later, this fact is in direct correlation with the positive perception generated in the participants who played the 2D version of this game about computer assisted instructions. This clearly reflects that players of the 2D version of this game enjoyed playing this version more in comparison to the participants who played the other two versions.

Table 7. Comparative table of the evaluation of computer assisted instruction before and after playing a version of "Outbreak:Safety First".

Versions		*Valuable*	*Useful*	*Comfortable*	*Pleasant*	*Stimulating*
	SCALE					
2D-pre	**1 2 3 4 5 6 7**	2.80	2.90	3.20	2.80	2.30
2D-post		2.70	2.50	3.00	2.60	2.40
3D-pre		3.00	3.12	2.94	3.24	3.18
3D-post		3.71	3.29	3.53	3.65	3.06
Mob-pre		3.11	3.00	4.33	3.56	3.22
Mob-post		3.44	3.33	3.22	3.56	3.11
		Worthless	*Useless*	*Uncomfortable*	*Unpleasant*	*Boring*

During the experiments, once all the participants were done playing their assigned version of this game, they were asked to report their feelings about computer-assisted instruction in a post game questionnaire. The pre- and post-game questionnaires shared a common rating table that allowed us to directly compare their answers before and after playing the game. The participants evaluated a set of 14 different items in the scale from 1(high) to 7(low), each of them related to a particular feeling about computer-assisted instruction. In an early analysis of the aggregated collected data from this table, and according to a subset of data included in Table 7, it is very interesting to see that the 2D version of this game qualified as the most valuable, useful, comfortable, pleasant, and stimulating across all three versions. According to our collected data, this positive evaluation of the 2D version extends to all other 9 questions about computer-assisted instruction. The data suggests that there is a correlation between the number of questions attempted in the 2D version and the positive feelings generated from playing this version of the game, and we infer that the level of enjoyment when playing the 2D version was higher compared to the other two versions. Table 7 shows a subset of our aggregated collected data.

As of now, statistical tests are being performed in order to evaluate the statistical significance of these data across different versions of the game. Although, it is worth mentioning that at this point there seems to be a correlation between the generated positive perception in the participants from playing the 2D version of this game, and the high scores obtained by the participants involved in the same version. As of now, further analysis on the collected data is being performed.

As part of the post game questionnaires, another table was included in order for the participants to evaluate from 1(Strongly Disagree) to 5(Strongly Agree) a set of 21 questions that were related to the content included in the game and the game in general. These questions were categorized into five big sets (please refer to the appendix section to see details about these questions); for the sake of providing a preliminary evaluation of the fAARS platform as a potential educational tool, we will focus on one of the categories that we consider as more related to learning, which includes questions about whether the students found playing a version of the "Outbreak:Safety First" game as a useful learning experience. In these category, the participants were asked to report their agreement from 1 to 5 (with 1 being “complete disagreement” and 5 being “complete agreement”) the following statements: *"Overall, this was a useful learning experience, in that it added to my: (q1) textbook learning, (q2) classroom learning, and (q3) practical learning."* Table 8 presents an average set of evaluations that were provided to each of these questions.

Table 8. Comparative table about learning experiences using different versions of "Outbreak:Safety First".

Version	**q1(textbook)**	**q2(classroom)**	**q3(practical)**
2D Version	3.30	4.20	4.10
3D Version	3.29	3.76	3.94
Mobile Version	2.78	3.56	3.89

Table 8 shows that the 2D version of this game stands out among all three versions as the version that (was reported to have) provided the most useful learning experiences across the classroom and practical learning dimensions. Table 8 also shows that participants across different versions of this game considered playing this game as having higher added value to their practical learning, than their textbooks and classroom learning. This suggests that students experienced these games as part of a practical scenario where they were able to use their skills learned from their textbooks and their classroom.

Additionally, in order to validate that participants learned something from the games developed in the fAARS platform, we asked the participants to do a test about PPE procedures and the spread of infections before and after playing a version of the game. As of now, 30 participants have already completed the pre- and post-game tests (six have not yet done so). Early results from these tests suggest that students performed better in the post-game test after all game versions. Table 9 shows that fewer participants responded incorrectly after playing the game than before (please refer to the appendix section to see the details about these questions). These four questions were related to PPE procedures, information about which was acquired through the scenarios in the game. The pre and post game tests also contained three open questions, but we are still in the process of analyzing the answers to these questions.

Table 9. Number of participants that answered incorrectly any of the four multiple choice questions in the pre and post game tests.

Version	q1	q2	q3	q4
2D-pre	6	0	0	4
2D-post	3	0	0	2
3D-pre	9	1	3	5
3D-post	7	0	2	3
Mob-pre	6	1	1	2
Mob-post	3	1	0	1

These early results are evidence that indeed all game variants have enabled some degree of learning to the participants, since their performance after the game improved relative to their performance before the game (for all game versions).

4.2.6 Some Concerns Around Game Development and Deployment

Upon reflecting on our experience with the development and deployment of the above games, we came to recognize some further issues that we had not been part of our original research concerns. These issues are related to safety in gaming, ethical concerns, errors and faults in the system, and how to deal with spotty connectivity in different environments. We have some initial answers on how to deal with some of these issues, like safety concerns for example, but there are others that require further investigation in the future. In most cases, before game deployment, it is very important to take into account all kinds of safety concerns, particular with respect to outdoor gaming. In order to avoid any safety issues, the game creators should have previous experience with the space where the game takes place. This is to guarantee that players will not have any problems during game play. It is also very important to offer players options for game completion, in that way, players who are unable to reach specific locations or environments, have the opportunity to continue playing safely.

Another important family of issues evolve around the ethical concerns when it comes to MAARGs game play. We have learned that anonymity is a very important, especially when collaboration for game completion is part of the tasks in the game. With the exception of face-to-face collaboration, fictitious characters and names should be always used during the game, and everything that happens in the game should allow players to enjoy them in a responsible manner, and without including any kind of offending content as part of a narrative of a game. In general, a game should have a very clear boundary about when a game starts and when it ends. It is very important that a game notifies its user when the game is on, and when it is off. This is to avoid having games that can surreptitiously affect the life of players. These and other ethical issues will certainly be the matter of further research and experimentation with different type of games.

In another vein, errors and faults in the system and spotty connectivity during game play can generate confusion among players. This is an aspect that we have not had the opportunity to study in depth. This is because our games developed in fAARS have been deployed in controlled environments. We are certainly aware of the importance of having robust games that maintain a constant communication with the fAARS platform to avoid creating confusions in players, but also, to avoid safety and ethical issues. Outdoor and indoor environments are very difficult to control, especially when there is not any kind of supervision during game play. MAARGs players are very difficult to monitor due to the nature of pervasive games. This can also lead to privacy issues, which is another research opportunity in pervasive gaming that is currently very active. Clearly, this is a very good opportunity for researchers to explore and experiment with.

We believe that the deployment of more MAARGs in different types of indoor and outdoor environments will help us better understand games and their role in entertainment and education. This will enable us to create more robust, secure, and enjoyable MAARGs free of ethical and safety shortcomings.

Chapter 5: Conclusion

Our work is still in its infancy; the fAARS platform can support a large variety of games only few of which we have explored and there are many still unanswered questions regarding the educational benefits of game-based experiences in the context of learning. Nevertheless, our experience with fAARS is very promising. More generally, we believe that the deployment of MAARGs for educational purposes is inevitable given the current momentum of the *"gamification"* paradigm. This is why developing robust systems and systematic methodologies for their deployment for different purposes is an extremely important and timely research agenda.

5.1 Contributions

In this thesis we have presented the fAARS platform that supports the development of a variety of MAARGs in the virtual treasure hunt and RPG styles. Games deployed in this platform can be played in different game-play modalities, and can be extended to virtual worlds where players can also interact with the games and other players. This notion of extending games to virtual worlds allows for the development of games with a more interesting narrative than typical desktop video games. Their potential of this type of games for education purposes makes them very interesting to educators in different areas. In general, our work makes the following contributions to the area of pervasive gaming:

1. ***The implementation of the fAARS platform***. The fAARS platform has been implemented in event-driven SOA architectural style, which supports peer-to-peer interactions in a virtual world, while maintaining the state of the game in a centralized server using a client-server communication style. The functionality of the platform has been wrapped within a set of RESTful APIs, through which communication of internal and external components of the platform is performed. The downstream functionality of the platform can be

extended by allowing external systems to register as observers of games in fAARS, and to push *Events* to the Game Engine if required.

2. ***The development of two games using the platform.*** Until now, two games have been developed in fAARS as part of two experiments performed in collaboration with the Psychology Department and Dr. Sarah Forgie from the Faculty of Medicine, in the University of Alberta. The first game titled ”Human Geometric Orientation”, was entirely developed and played in a virtual world, where participants where trained and tested to identify a set of geometric features in a number of parallelogram-shaped rooms. The second game called "Outbreak:Safety First", exploits the capabilities of the fAARS platform by deploying three versions of the same game in two different game play modalities. The objective of this experiment was to test the platform as a potential educational tool.
3. ***Preliminary experimental results demonstrate the educational potential that the fAARS platform offers to educators.*** The versatility in terms of the game play modalities that are supported by fAARS allows educators to embed educational content in different dimensions and through narratives that are impossible in regular desktop video games. Early results from the experiments performed using the game "Outbreak:Safety First" demonstrate that games deployed in fAARS can be used as a medium to provide educational content within the context of a class while players learn in a fun way. The important aspect of the games developed in fAARS for educational purposes is the creativity in which the class content is embedded as part of the story of a game.

5.2 Future Work

In the future, we plan to further develop the platform to support the development and deployment of a larger set of MAARGs. This objective can be achieved by extending the supported set of interactions among players in a game in the real and virtual worlds.

Second, we intend to extend the peer-to-peer interactions among players in the real world also, through the usage of more advance smart-phone sensors such as Bluetooth and NFC.

Third, a new set of web services will be implemented in order to include a wider spectrum of input sources of data of different types. This will enable the platform to support different kinds of game-play experiences in later releases of the platform, and to enrich the narratives of games through the inclusion of other types of information. This can be done by defining extra parameters to the current RESTful interface, which will support the provision of further details about *Events* pushed to the fAARS Game Engine.

Finally, we plan to integrate an authoring environment through which game creators and educators can create games.

In parallel, we intend to be continuously developing specific games on fAARS experimenting with different knowledge domains, different type sof knowledge and skills, and different types of games and game-play modalities. Through this systematic experimentation, we want to study in depth the role of games in learning: when they work and when they don't and what a checklist might be for recognizing an appropriate game for integration in a given curriculum.

The future of pervasive gaming is promising, specifically for MAARGs, which are undoubtedly becoming in important educational tools, which can help to further develop a specific type of skill, and to encourage physical activity while playing a fun game and learning something in the process. The development of the fAARS platform is one of the first steps towards that future.

REFERENCES

[1] Lindley, C. A. Trans-reality gaming. *In Proceedings of the 2nd annual International Workshop in Computer Game Design and Technology* (Liverpool, UK: Liverpool John Moores University, 2004), pp. 15-16.

[2] IEEE Spectrum. Special report: top 11 technologies of the decade. *In IEEE Spectrum Inside Technology*. http://spectrum.ieee.org/static/special-report-top-11-technologies-of-the-decade (accessed February 2011)

[3] Lin, F., and Ye, W. Operating system battle in the ecosystem of smart-phone industry. *In Proceedings of the 2009 International Symposium on Information Engineering and Electronic Commerce* (Washington, DC, USA, 2009), IEEE Computer Society, pp. 617-621.

[4] Barkhuus, L., and Dey, A. K. Location-based services for mobile telephony: a study of users' privacy concerns. *In INTERACT* (2003), pp. 709-712. M. Rauterberg, M. Menozzi, and J. Wesson, Eds., IOS Press.

[5] Kulkarni, D., and Tripathi, A. A framework for programming robust context-aware applications. *IEEE Trans. Softw. Eng. 36* (March 2010), pp. 184-197.

[6] Naguib, H., Coulouris, G., and Mitchell, S. Middleware support for context-aware multimedia applications. *In Proceedings of the IFIP TC6 / WG6.1 Third International Working Conference on New Developments in Distributed Applications and Interoperable Systems* (Deventer, The Netherlands, The Netherlands, 2001), Kluwer, B.V., pp. 9-22.

[7] Schiller, J. H., and Voisard, A., Eds. *Location-Based Services*. Morgan Kaufmann, 2004.

[8] Steiniger, S., Neun, M., and Edwardes A. Foundations of location-based services. *CartouCHe Lecture Notes on LBS, version 1.0.* (Department of Geography, University of Zurich, Switzerland, 2006), pp. 1-28.

[9] Magerkurth, C., Cheok, A.D., Mandryk, R.L., and Nilsen, T. Pervasive games: Bringing computer entertainment back to the real world. *ACM Computers in Entertainment*, *3,*3, (2005), Article 4A.

[10] Montola, M. Exploring the edge of the magic circle: Defining pervasive games, vol. 1966. *Citeseer* (2005), pp. 16-19.

[11] Martin, A., Thompson, B., and Chatfield, T. Alternate Reality Games White Paper - *IGDA ARG SIG. International Game Developers Association* (Mt. Royal, New Jersey, 2006).

[12] Stroulia, E. Smart services across the real and virtual worlds. *In The Smart Internet* (Springer-Verlag, Berlin, Heidelberg, 2010), pp. 178-196.

[13] Piekarski, W., and Thomas, B. Arquake: the outdoor augmented reality gaming system. *Commun. ACM 45* (January 2002), pp. 36-38.

[14] Cheok, A. D., Goh, K. H., Liu, W., Farbiz, F., Fong, S. W., Teo, S. L., Li, Y., and Yang, X. Human pacman: a mobile, wide-area entertainment system based on physical, social, and ubiquitous computing. *Personal Ubiquitous Comput. 8* (May 2004), pp. 71-81.

[15] Qui, T. C. T., Nguyen, T. H. D., Mallawaarachchi, A., Xu, K., Liu, W., Lee, S. P., Zhou, Z. Y., Teo, S. L., Teo, H. S., Thang, L. N., Li, Y., Cheok, A. D., and Kato, H. Magic land: live 3d human capture mixed reality interactive system. *In CHI '05 extended abstracts on Human factors in computing systems* (New York, NY, USA, 2005), CHI EA '05, ACM, pp. 1142-1143.

[16] Benford, S., Crabtree, A., Flintham, M., Drozd, A., Anastasi, R., Paxton, M., Tandavanitj, N., Adams, M., and Row-Farr, J. Can you see me now? *ACM Trans. Comput.-Hum. Interact. 13* (March 2006), pp. 100-133.

[17] Benford, S., Flintham, M., Drozd, A., Anastasi, R., Rowland, D., Tandavanitj, N., Adams, M., Row-Farr, J., Oldroyd, A. and Sutton, J. Uncle Roy All Around You: Implicating the City in a Location-Based Performance. In *Proc. Advances in Computer Entertainment (ACE 2004)*, ACM Press.

[18] Flintham, M., J. Humble, N. Tandavanitj, M. Adams, J. Row Farr. I Like Frank: a mixed reality game for 3G phones. Submitted to *IEEE Computer Graphics And Applications*.

[19] Lubyk, DM., Dupuis, B., Gutierrez, L., and Spetch, M. Geometric orientation by humans: angles weigh in. Submitted to *Psychonomic Bulletin & Review* (Oct. 11, 2011). Manuscript ID: PBR-BR-11-323.

[20] Fitz-Walter, Z., and Tjondronegoro, D. Exploring the opportunities and challenges of using mobile sensing for gamification. *Ubicomp 2011 Workshop. Mobile Sensing: Challenges, Opportunities and Future Directions* (Beijing, China, 2011).

[21] Lindley, C. A. Game space design foundations for trans-reality games. *In Proceedings of the 2005 ACM SIGCHI International Conference on Advances in computer entertainment technology* (New York, NY, USA, 2005), ACE '05, ACM, pp. 397-404.

[22] Benford, S., Magerkurth, C., and Ljungstrand, P. Bridging the physical and digital in pervasive gaming. *Commun. ACM 48* (March 2005), pp. 54-57.

[23] Wagner, D., and Schmalstieg, D. History and future of tracking for mobile phone augmented reality. *In Proceedings of the 2009 International Symposium on Ubiquitous Virtual Reality* (Washington, DC, USA, 2009), IEEE Computer Society, pp. 7-10.

[24] Lindt, I., Broll, W.: NetAttack – First Steps Towards Pervasive Gaming, *ERCIM NEWS Special Issue on Games Technology, No. 57* (April 2004,), pp. 49-50.

[25] Lindt, I., Ohlenburg, J., Pankoke-Babatz, U., Prinz, W., and Ghellal, S. Combining multiple gaming interfaces in epidemic menace. *In CHI '06 extended abstracts on Human factors in computing systems* (New York, NY, USA, 2006), CHI EA '06, ACM, pp. 213-218.

[26] Lindt, I., Ohlenburg, J., Pankoke-Babatz, U., Ghellal, S., Oppermann L., and Adams, M. Designing Cross Media Games. *In Proc. PerGames Workshop at Pervasive* (2005).

[27] Herbst, I., Braun, A.-K., McCall, R., and Broll, W. Timewarp: interactive time travel with a mobile mixed reality game. *In Proceedings of the 10th international conference on Human computer interaction with mobile devices and services* (New York, NY, USA, 2008), MobileHCI '08, ACM, pp. 235-244.

[28] O'Shea, P. M., Mitchell, R., Johnston, C., and Dede, C. J. Lessons learned about designing augmented realities. *IJGCMS 1, 1* (2009), pp. 1-15.

[29] Chang, W.-C., Wang, T.-H., Lin, F. H., and Yang, H.-C. Game-based learning with ubiquitous technologies. *IEEE Internet Computing 13* (July 2009), pp. 26-33.

[30] Milgram, P., Takemura, H., Utsumi, A., and Kishino, F. Augmented reality: a class of displays on the reality-virtuality continuum. *SPIE Telemanipulator and Telepresence Technologies 2351, Telemanipulator and Telepresence Technologies* (1994), pp. 282-292.

[31] Gao, J. Hybrid tracking and visual search. *In Proceedings of the 16th ACM international conference on Multimedia* (New York, NY, USA, 2008), MM '08, ACM, pp. 909-912.

[32] Honkamaa, P., Siltanen, S., Jäppinen, J., Woodward, C., Korkalo, O. Interactive outdoor mobile augmentation using markerless tracking and GPS, *Proc. Virtual Reality International Conference (VRIC)* (Laval, France, April 2007), pp. 285-288.

[33] Zendjebil, I. M., Ababsa, F., Didier, J.-Y., and Mallem, M. On the hybrid aid-localization for outdoor augmented reality applications. *In Proceedings of the 2008 ACM symposium on Virtual reality software and technology* (New York, NY, USA, 2008), VRST '08, ACM, pp. 249-250.

[34] Mottola, L., Murphy, A. L., and Picco, G. P. Pervasive games in a mote-enabled virtual world using tuple space middleware. *In Proceedings of 5th ACM SIGCOMM workshop on Network and system support for games* (New York, NY, USA, 2006), NetGames '06, ACM.

[35] Tutzschke, J.-P., and Zukunft, O. Frap: a framework for pervasive games. *In Proceedings of the 1st ACM SIGCHI symposium on Engineering interactive computing systems* (New York, NY, USA, 2009), EICS '09, ACM, pp. 133-142.

[36] Broll, W., Ohlenburg, J., Lindt, I., Herbst, I., and Braun, A.- K. Meeting technology challenges of pervasive augmented reality games. *In Proceedings of 5th ACM SIGCOMM workshop on Network and system support for games* (New York, NY, USA, 2006), NetGames '06, ACM.

[37] Lindley, C. and Eladhari, M. Narrative structure in trans-reality role-playing games: Integrating story construction from live action, table top and computer based role-playing games. *In Proceedings of DIGRA 2005* (Vancouver, Canada, 2005).

[38] Web Article. Pac-Manhattan. *The Pacmanhattan. Project Website*. http://pacmanhattan.com/about.php (accessed January 2011).

[39] Michelson, B. Event-Driven Architecture Overview, Event-Driven SOA is just part of the EDA Story. *Patricia Seybold Group 2006*. http://www.omg.org/soa/Uploaded%20Docs/EDA/bda2-2-06cc.pdf (accessed March 2011).

[40] Spelke, E., Lee, S. A., and Izard, V. Beyond core knowledge: Natural geometry. *Cognitive Science 34, 5* (2010), pp. 863-884.

[41] Mansour, S. and El-Said, d M. Multi-Players Role- Playing Educational Serious Games: A Link between Fun and Learning. *The International Journal of Learning* (2008), 15(11), pp. 229-240.

[42] Skiba, D. J., and Barton, A. J. Adapting your teaching to accommodate the net generation of learners. *Online Journal of Issues in Nursing* 11, 2 (2006), 5.

[43] Kron, F. W., Gjerde, C. L., Sen, A., and Fetters, M. D. Medical student attitudes toward video games and related new media technologies in medical education. *BMC Medical Education* (2010), 10, 50-50.

[44] Boyce, JM. Pittet,D. Healthcare Infection Control Practices Advisory Committee, HICPAC/SHEA/APIC/IDSA Hand Hygiene Task Force: Guideline for Hand Hygiene in Health-Care Settings. Recommendations of the Healthcare Infection Control Practices Advisory Committee and the HICPAC/SHEA/APIC/IDSA Hand Hygiene Task Force. *Society for Healthcare Epidemiology of America/Association for Professionals in Infection Control/Infectious Diseases Society of America*. MMWR Recomm Rep 2002 Oct 25;51(RR-16):1–45, Oct. 2002.

[45] Pittet, D. Improving compliance with hand hygiene in hospitals. *Infection control and hospital epidemiology the official journal of the Society of Hospital Epidemiologists of America* 21, 6 (2000), pp. 381-386.

[46] Association of Medical Microbiology and Infectious Diseases - Canada. Public Reporting and Inter-hospital Comparison of Health Care-Acquired Infections. *In Community and Hospital Infection Control Association - Canada*. http://www.chica.org/pdf/AMMIposition.pdf (accessed March 2011).

[47] Web Article. Emerging Infection Deseases. *In Centers for Disease Control and Prevention*. http://www.cdc.gov/ncidod/eid/vol4no3/weinstein.htm (accessed March 2011).

[48] Gould, D. J., Chudleigh, J. H., Moralejo, D., and Drey, N. Interventions to improve hand hygiene compliance in patient care. *Cochrane database of systematic reviews Online* 68, 3 (2007), CD005186.

[49] Pittet, D., Hugonnet, S., Harbarth, S., Mourouga, P., Sauvan, V., Touveneau, S., and Perneger, T. V. Efectiveness of a hospital-wide programme to improve compliance with hand hygiene. Infection control programme. *Lance 356*, 9238 (2000), pp. 1307-1312..

[50] Gutierrez, L., Nikolaidis, I., Stroulia, E., Gouglas, S., Rockwell, G., Boechler, P., Carbonaro, M., and King, S. fAR-Play: A framework to develop augmented/alternate reality games. *In PerCom Workshops (2011), IEEE*, pp. 531-536.

[51] SCVNGR Team. SCVNGR. *The SCVNGR Team Business Website*. http://www.scvngr.com/ (accessed July 2011)

[52] Imajie Project. Playar. *The Playar Project Website*. http://code.google.com/p/playar/ (accessed September 2011)

[53] Suomela, R., Räsänen, E., Koivisto, A., and Mattila, J. Open-Source Game Development with the Multi-user Publishing Environment (MUPE) Application Platform. *In: ICEC Bd. 3166*, Springer, 2004, pp. 308--320.

[54] Fetter, M., Etz, M., and Blechschmied, H. Mobile chase--towards a framework for location-based gaming. *In: GRAPP (AS/IE), INSTICC--Institute for Systems and Technologies of Information*, Control and Communication, 2007, pp. 98--105

[55] GroudSpeak. Wherigo. *The Wherigo Beta Project Website*. http://www.wherigo.com/ (accessed September 2011)

[56] Saha, D., and Mukherjee, A. Pervasive computing: A paradigm for the 21st century. *Computer 36* (March 2003), pp. 25-31.

[57] Roussos, G., Marsh, A. J., and Maglavera, S. Enabling pervasive computing with smart phones. *IEEE Pervasive Computing 4* (April 2005), pp. 20-27.

[58] Rashid, O., Bamford, W., Coulton, P., Edwards, R., and Scheible, J. Pac-lan: mixed-reality gaming with rfid-enabled mobile phones. *Comput. Entertain. 4* (Oct. 2006).

[59] A Different Game. Ghostwire. *The Ghostwire Promotional Website*. http://www.ghostwiregame.com/ (accessed January 2012)

[60] Electronic Arts Inc. NBA Live 365. *The NBA Live 365 Promotional Website*. http://www.ea.com/nba-live?platform=live09 (accessed January 2012)

[61] Cogbill, D., Endo, Y., and Malik, R. Weakness. *The Weakness Project Website*. http://mobile.parsons.edu/weekness/ (accessed January 2012)

[62] 42 Entertainment. I Love Bees. *The I Love Bees Project Website*. http://ilovebees.com/ (accessed January 2012)

[63] 42 Entertainment. Year Zero. *The Year Zero Project Website*. http://yearzero.nin.com/ (accessed January 2012)

[64] World Bank Institute. EVOKE. *The EVOKE Project Website*. http://www.urgentevoke.com/ (accessed January 2012)

[65] ITVS Interactive. World Without Oil. *The World Without Oil Project Website*. http://www.worldwithoutoil.org/ (accessed January 2012)

[66] Papakonstantinou, S., and Brujic-Okretic, V. Prototyping a context-aware framework for pervasive entertainment applications. *In Proceedings of the 2009 Conference in Games and Virtual Worlds for Serious Applications* (Washington, DC, USA, 2009), VS-GAMES '09, IEEE Computer Society, pp. 84-91.

[67] Ganev, V., Chodos, D., Nikolaidis, I., and Stroulia, E. The smart condo: integrating sensor networks and virtual worlds. *In Proceedings of the 2nd Workshop on Software Engineering for Sensor Network Applications* (New York, NY, USA, 2011), SESENA '11, ACM, pp. 49-54.

[68] Coronato, A., and De Pietro, G. A web services based architecture for supporting mobile users in large enterprises. *J. Web Eng. 6* (June 2007), pp. 131-142.

[69] Ishii, H., Mazalek, A., and Lee, J. Bottles as a minimal interface to access digital information. In CHI '01 extended abstracts on Human factors in computing systems (New York, NY, USA, 2001), CHI EA '01, ACM, pp. 187-188.

APPENDIX

1. fAARS RESTful API

Table 10. fAARS RESTful API

Event-driven Machine			
API Name	***Functional Description***	***Parameters***	***Response***
catchEvents	Notify the internal Event-driven Machine about a new incoming *Event* during game play. The Event-driven Machine will then process the corresponding *ECA* game rules of a game.	[eventGeneratorID, eventName, eventRecipientID, gameID]	Success or, Error
Virtual World			
API Name	***Functional Description***	***Parameters***	***Response***
rezObject	Create a new virtual *Actor* in the virtual world.	[vwSimulationID, agentID]	Success or, Error
killObject	Delete a virtual *Actor* in the virtual world.	[vwSimulationID, agentID]	Success or, Error
setLocationByID	Update location of a	[vwSimulationID,	Success or,

	virtual *Actor* by location ID.	agentID, locationID]	Error
setLocationByGPS	Update location of a virtual *Actor* by GPS coordinates.	[vwSimulationID, agentID, latitude, longitude]	Success or, Error
setLocationByVector	Update location of a virtual *Actor* by <x,y,z> coordinates.	[vwSimulationID, agentID, xcoord, ycoord, zcoord]	Success or, Error
setCollisionEvent	Turn on/off the collision detection sensor of a virtual *Actor*.	[vwSimulationID, agentID, active]	Success or, Error
setAtTargetEvent	Turn on/off the change-of-location sensor of a virtual *Actor*.	[vwSimulationID, agentID, active]	Success or, Error
setOnProximityEvent	Turn on/off the proximity sensor of a virtual *Actor*.	[vwSimulationID, agentID, active]	Success or, Error
setMobilitySimulator	Set the mobility simulator engine that will update the location of a virtual *Actor* on demand.	[vwSimulationID, agentID, URL]	Success or, Error
setURLPushEvents	Set the Event-driven Machine URL to which	[vwSimulationID, agentID, URL]	Success or, Error

	a virtual *Actor* should notify about detected events.		
Game Manager			
API Name	***Functional Description***	***Parameters***	***Response***
addUser	Add a new user as an *Actor* to the pool of *Actors* in the game..	[gameID, username, password, email]	Success or, Error
activateUser	Activate/Deactivate an *Actor* in a game.	[gameID, userID, active]	Success or, Error
loginUser	Logs in an *Actor* using username and password.	[gameID, username, password]	[userID] or, Error
logoffUser	Logs off an *Actor*.	[gameID, userID]	Success or, Error
consentUser	A user consents to share information during game play. This information includes location, actions, and events generated.	[gameID, consentID, userID, consent]	Success or, Error
getPlayers	Get the list of *Actor* in a game, which includes information about each	[gameID]	[list_of_players] or, Error

	Actor such as username, userID, active, status, score.		
getGameMetadata	Get information about the game: number of *Actor*, game status(on/off), time running, URLsimulator, vwSimulatorID, among other details.	[gameID]	[metadata_info] or, Error
startGame	Sets the state of a game to ON, which allows *Actor* log in.	[gameID]	Success or, Error
stopGame	Sets the stat of a game to OFF, which causes *Actor* not to be able to log in.	[gameID]	Success or, Error
getScores	Get a list of scores of all *Actors* in a game.	[gameID]	[list_scores] or, Error
getStates	Get a list of states of all *Actors* in a game.	[gameID]	[list_states] or, Error
getPlayerScore	Get the score of one *Actor* in a game.	[gameID, userID]	[player_score] or, Error
getPlayerState	Get the current state of one *Actor* in a game.	[gameID, userID]	[player_state] or, Error
getConsentFormID	Get the consent form identifier attached to a	[gameID]	[consentID] or,

	game for data collection of players.		Error
uploadFile	Upload a file to the fAARS repository.	[fileName]	[URL] or, Error
Message Manager			
registerObserverEPE	Add an *Actor* to the list of observers of the *Event* Processing Engine.	[clientID]	Success or, Error
registerSubscriber	Add an external system to the list of observers of a game running in the fAARS platform. This allows to notify registered Event Subscribers about game *Events*.	[clientKey, gameID]	Success or, Error

As shown in Table 10, these APIs can return either a "Success" or "Error" response. On either type of response, the message is always contained in a JSON file. We decided to use JSON files as part of the message transmission medium because this is a lightweight and the most appropriately method for decoding messages in a natural way using Javascript objects on the client side. As seen in Table 10, most of these APIs return an "OK" acknowledgement on success, which means that the API has received and processed the requested service successfully. In cases when an API returns data as part of the "OK" acknowledgement, everything is included together in a JSON file. In cases when the API is unable to perform the requested service, the response will include an error number plus an error message, so the client can act accordingly. This is assuming that the service is up and running, and that the error is caused by an internal process in fAARS

2. "Outbreak:Safety First" Experiment Documents

2.1 Consent Form

Research Information and Participants' Consent Form

Assessing the Educational Impact of Course Gamification (in Epidemiology)

Purpose. You are invited to participate in a research study, *Assessing the Educational Impact of Course Gamification (in Epidemiology)*, being conducted by Dr. Eleni Stroulia and M.Sc. student Lucio Gutierrez (Department of Computing Science), and Sarah Forgie (Faculty of Medicine), of the University of Alberta. This study examines the effectiveness of serious games and platforms for education.

Your participation. Your participation in this study will occur in three phases. In the first phase, you will take a short quiz, and fill out a questionnaire. After that, you will receive instructions on how to play a game. In the third phase, you will take a follow-up quiz, and fill out a second questionnaire. The first and third phases will take up to 45 minutes together, while the second phase may take up to an hour. Thus, the total participation time is expected to be no more than two hours in total.

During participation. During the game play of each of these games, the game engine will record the communication and interactions among you and other players, including:

a) what messages you send to other players using which channel;
b) what you do in the real world, including taking pictures, scanning QR tags, being close and/or following other players;
c) what you do in the virtual world, including your choices of movements, gestures, accessories and relative proximity to other players;
d) how you use the user interface, including checking game statistics, and accessing game information; and
e) what is your location in the real world and/or the virtual world at all time.

At the end of the game, you will be asked to fill out an online questionnaire about your experience with the game and your opinion of the usability of the mobile application in the real world and the user interface in the virtual world.

Data collected through each of these games can give us insights on many interesting research problems including (a) patterns in how people interact and network in these events, (b) models of pedestrian mobility and (c) the longer-term impact of such games to the extent that they can be adopted to be used as regular tools for learning. These data will be kept for a minimum of 5 years following completion of the research study and will remain in secure storage inside the firewall of the Computing Science department. You are entitled to a copy of the final report of this study. Results from this research study may be used for research articles, presentations, and teaching purposes. For all uses, data will be fully anonymized and handled in compliance with the University Standards. Other research assistants may have access to the anonymized data for analysis purposes.

Your rights. There are several very clear rights that you are entitled to as a participant in any research conducted by a researcher from the University of Alberta. You have the right:

- To not participate.
- To withdraw at any time without prejudice to pre-existing entitlements, and to continuing and meaningful opportunities for deciding whether or not to continue to participate.
- To opt out without penalty and any collected data withdrawn from the database and not included in the study.
- To privacy, anonymity and confidentiality.

- To safeguards for security of data (data are to be kept for a minimum of 5 years following completion of research).
- To disclosure of the presence of any apparent or actual conflict of interest on the part of the researcher(s).
- To a copy of any final report that may be a result of the collected data.

Questions about your rights as a research participant may be directed to the University of Alberta Research Ethics Office at telephone number (780) 492-2615.

Feel free to contact Dr. Eleni Stroulia (stroulia@ualberta.ca) and M.Sc. student Lucio Gutierrez at 780-716-7592 or at (lucio@ualberta.ca) if you have any questions or comments.

Benefits and risks. This research can potentially contribute to the development of better ways of delivering simulation-based serious games, tools, and platformss. There are no foreseeable risks to this study.

Signatures. Please write your name, indicate your year of studies, gender, age, e-mail address, and sign below to indicate that you have read and understood the nature and purpose of the study. Your signature acknowledges the receipt of a copy of the consent form as well as indicates your willingness to participate in this study.

1. Name: ____

2. Year of studies : 1st year / 2nd year / 3rd year / 4th year / Other: ____________

3. Gender: Male / Female

4. Age: _______________________

5. E-mail address: ____

____ ______________________

Participant's Signature Date

____ ______________________

Researcher's Signature Date

2.2 Pre and Post Game Test

Enter your assigned participant ID here:________________________

1. A four year old child is admitted to the hospital with whooping cough. As you perform hand hygiene, you think about how *Bordetella pertussis* is spread. You remember it is spread by droplets only. What personal protective equipment will you put on before entering this child's room and examining her? (Please check all that apply)

! Gown

! Procedure mask

! N-95 respirator

! Eye protection

! Gloves

2. You are examining an eighteen year old with suspected parainfluenza virus. She is having frequent bouts of coughing. You are happy that you remembered to put on all of the equipment necessary for droplet and contact precautions, and your risk of acquiring this virus is dramatically reduced. What personal protective equipment are you wearing? (Please check all that apply)

! Gown

! Procedure mask

! N-95 respirator

! Eye protection

! Gloves

3. Mr. S is in the emergency room with bloody diarrhea. There is a large sign on the door saying "contact precautions." Before entering his examination room you perform hand hygiene and then don your personal protective equipment. You want to be careful because you really don't want to catch diarrhea from this patient. What personal protective equipment is needed for contact precautions? (Please check all that apply)

! Gown

! Procedure mask

! N-95 respirator

! Eye protection

! Gloves

4. Mr S. gets admitted to the ward and you are pleased that you have a few minutes to eat your lunch and check your emails. Since Mr. S's room is now empty, you dash in to eat and use the computer. You washed your hands before eating to be extra careful. You finish your shift in the Emergency Room and go home. Within 12 hours you have a fever and abdominal pain. Bloody diarrhea ensues. Mr. S's diagnosis if *Shigella dysenteriae*. You washed your hands before you ate and took great care with your personal protective equipment…so how did you get infected with Shigella? (Please write your answer below)

____
____

5. Name three infections that require airborne precautions? (Please write your answers below)

a)

b)

c)

6. What PPE is required for a person on airborne precautions? (Please check all that apply)

! Gown

! Procedure mask

! N-95 respirator

! Eye protection

! Gloves

7. Please write the correct order for removing (doffing) your personal protective equipment (assume you are wearing eye protection, gloves, gown, mask) and performing hand hygiene. (Each letter can be used once, more than once, or not at all).

a) Gown

b) Procedure mask

c) Eye protection

d) Gloves

e) Hand hygiene

____

End of test – thank you!

2.3 Pre Game Questionnaire

****Enter your assigned participant ID here:*** ________________________

Briefly comment on your expectations for this simulation:

1. Rate your previous experience with educational simulations/games.

None 1 2 3 4 5 Extensive

What types of educational simulations/games have you had experience with?

____
____
____
____

2. How do you rate your previous experience with virtual worlds and/or online environments. (Instead of *virtual worlds*, the term *location-based games using mobile phones* will be used for the mobile version group)

None 1 2 3 4 5 6 7 Extensive

With which virtual worlds or simulations (Instead of *virtual worlds*, the term *location-based games using mobile phones* will be used for the mobile version group) are you familiar and how long have you spent in them to date?

____
____
____
____

3. What are your expectations for this simulation/game?

Low/Pessimistic 1 2 3 4 5 6 7
High/Optimistic

What, in particular, do you expect?

Using the following chart, indicate your feelings about computer assisted instruction. This is not a test; there are no right or wrong answers. The purpose of the items in the chart below is to measure the meaning that the idea of “computer aided instruction” has for you. For each item in the chart, if you feel that computer aided instruction is very closely related to one end of the chart, then place an X at one end of the chart. If you feel that the item is neutral or irrelevant, then place an X in the middle.

Make an independent judgement on each descriptive scale. Do not try to remember how you marked similar items. Work at a fairly high speed, recording your first impression or feeling about an item. Do not skip any items. Do not put more than one mark on a single adjective scale.

Rigid								Flexible
Useful								Useless
Stimulating								Boring
Meaningless								Meaningful
Pleasant								Unpleasant
Valuable								Worthless
Creative								Unimaginative
Impersonal								Personal
Efficient								Inefficient
Inappropriate								Appropriate
Comfortable								Uncomfortable
Non-threatening								Threatening
Overpowering								Easy to control
Timesaving								Time-consuming

2.4 Post Game Questionnaire

***Enter your assigned participant ID here:** ________________________

Satisfaction Questionnaire	**SD**	**D**	**N**	**A**	**SA**	**N/A**
Using the following scale, please rate your satisfaction with each of the following aspects of this simulation	strongly disagree	disagree	neutral	agree	strongly agree	not applicable
1. This simulation has enhanced my understanding of the personal precautions that I should take to assist patients.	SD	D	N	A	SA	NA
2. The level of realism was sufficient for **suspension of disbelief** in the following areas:						
a. The scenarios presented	SD	D	N	A	SA	NA
b. Wearing the proper medical equipment	SD	D	N	A	SA	NA
c. How infections are spread	SD	D	N	A	SA	NA
d. The hospital department	SD	D	N	A	SA	NA
3. The level of realism was sufficient to **enable learning** in the following areas:						
a. The right order to wear the proper equipment	SD	D	N	A	SA	NA
b. Understand the importance of wearing the proper equipment	SD	D	N	A	SA	NA
c. When it is important to wear the proper equipment						
d. The importance of washing my hands						
4. I was able to apply the following knowledge and skills in completing the scenario:						
a. Knowledge of medical facts	SD	D	N	A	SA	NA
b. Knowledge of relevant procedures	SD	D	N	A	SA	NA
c. Communication skills.	SD	D	N	A	SA	NA

5. Throughout the simulation, I understood how to play the game during any given situation.	SD	D	N	A	SA	NA
6. The simulation was interesting, and I felt engaged in the experience.	SD	D	N	A	SA	NA
7. The scenario was well organized.	SD	D	N	A	SA	NA
8. The following activities were useful in facilitating my learning:						
a. Pre-simulation instructions	SD	D	N	A	SA	NA
b. Pre-simulation practice	SD	D	N	A	SA	NA
c. Simulation experience	SD	D	N	A	SA	NA
9. I would recommend this simulation to other students.	SD	D	N	A	SA	NA
10. Overall, this was a useful learning experience, in that it added to my						
a. Textbook learning	SD	D	N	A	SA	NA
b. Classroom learning	SD	D	N	A	SA	NA
c. Practical learning	SD	D	N	A	SA	NA

Briefly comment on your experiences in this simulation:

1. What did you like about this simulation/game?

____
____
____
____
____

2. What would you change or improve?

____
____
____
____
____

3. What did you learn about personal precautions when treating patients?

____

4. What was your experience with the interface of this simulation?

____

5. Other comments?

____

Using the following chart, indicate your feelings about computer assisted instruction. This is not a test; there are no right or wrong answers. The purpose of the items in the chart below is to measure the meaning that the idea of “computer aided instruction” has for you. For each item in the chart, if you feel that computer aided instruction is very closely related to one end of the chart, then place an X at one end of the chart. If you feel that the item is neutral or irrelevant, then place an X in the middle.

Make an independent judgement on each descriptive scale. Do not try to remember how you marked similar items. Work at a fairly high speed, recording your first impression or feeling about an item. Do not skip any items. Do not put more than one mark on a single adjective scale.

Rigid								Flexible
Useful								Useless
Stimulating								Boring
Meaningless								Meaningful
Pleasant								Unpleasant
Valuable								Worthless
Creative								Unimaginative
Impersonal								Personal
Efficient								Inefficient
Inappropriate								Appropriate
Comfortable								Uncomfortable
Non-threatening								Threatening
Overpowering								Easy to control
Timesaving								Time-consuming