Addressing Openness and Portability in Outdoor Pervasive Role-Playing Games

Vlasios Kasapakis, Damianos Gavalas, Nikos Bubaris
Department of Cultural Technology and Communication
University of the Aegean
Mytilene, Greece
{v.kasapakis, dgavalas}@aegean.gr, nbubaris@ct.aegean.gr

***Abstract*—*This article introduces Barbarossa, an outdoor pervasive role-playing game. Existing pervasive game prototypes do not enable relocation of the game space as they heavily rely in orchestration actions. They also overlook several aspects which critically affect user acceptance and game experience such as scenario design, usability of employed technologies, game duration and intensity. Barbarossa addresses the abovementioned issues featuring portable game modes through moderating or seamlessly embedding any orchestration actions needed within the game process. It also takes into account concrete technology usage requirements for each game mode according to the game session duration and player effort required. Game experience is enhanced through incorporating several contextual parameters, while game rules may be personalized based on players' profile data retrieved from social networks.*

Keywords—*Pervasive games; role-playing; orchestration; openness; user-generated content.*

I. INTRODUCTION

Pervasive games extend the gaming experience out into the real world be it on city streets, in the remote wilderness or a living room. Players with mobile computing devices move through the game world, while sensors (either on-board or weaved into the game space) capture contextual information used to perform game activities that vary depending on where users are, what they do or even how they feel [1].

Pervasive games can be classified in sub-genres mainly based on their supported features and game environment. The most popular genres are *location-based games (*they take into account the player's changing relative or absolute position into the game rules) [2, 3], *trans-reality games* (they incorporate physical, virtual and mixed reality game staging spaces) [4-7] and *augmented and mixed-reality (AR, MR)* games (they seek to integrate virtual and physical elements within a coherently experienced perceptual game world) [7-9].

Many existing pervasive game prototypes have been evaluated through user trials, conveying useful conclusions with regards to technology usage and means to enhance players' immersion [3-6, 9].

Several well-known pervasive games exist, for instance CYSMN? (Can You See Me Now?) [6], URAAY (Uncle Roy All Around You) [5] , CTF (Capture The Flag) [4], EM II (Epidemic Menace II) [8] and TimeWarp [9]. Most of the abovementioned games are location-aware (location is commonly estimated by a GPS receiver [4, 6, 8] or a GPS receiver assisted by another technology like Computer vision [9]). In URAAY no GPS receiver is used and players define their own location using a map [5]. As for networking most of the games use GPRS technology [4, 5, 8] with CTF making also use of Bluetooth and EMII additionally using WLAN connectivity. Apart from location, most games also incorporate social context into their logic as the player is aware of other players' locations and activities [4-6, 8, 9].

Notably, most prototyped pervasive games often fall short with respect to various design/gaming aspects. Firstly, the game content and staging is rigidly defined at design time by the game designer/moderator. Hence, it becomes difficult to relocate the game space elsewhere, even more so to engage the users in game design and content editing; those issues severely restrict games' portability, variability and openness [10], while players' involvement is less participatory. Secondly, pervasive games typically entail considerable orchestration resources; those may include dedicated infrastructure deployed throughout the game space in support of game activities or individuals/actors recruited to monitor players' conformation to game rules or provide assistance. Thirdly, several critical pervasive game aspects such as the design of the actual game scenarios, the usability of employed technologies, the game's duration and intensity have been largely overlooked. We argue that an in-depth investigation of those aspects is of critical importance in order to substantiate game design decisions that will raise user acceptance and result in immersive game experiences.

In this paper we introduce *Barbarossa*, a trans-reality pervasive role-playing game, aiming at addressing the abovementioned shortcomings. Our focus is on designing open and portable game engine architecture to ease game deployment at any setting, while relaxing the requirements for human support (i.e. orchestration) in the game setup. We also feed players' social network profile data into the game rules to provide personalized game experiences.

978-1-4673-5307-6/13/$31.00 ©2013 IEEE

Three player roles are identified in *Barbarossa*, each following an individual game sub scenario; their common objective is to trace and open a chest, locked with a four digit combination lock and hidden somewhere in the city of Mytilene (Lesvos, Greece), in order to claim the rewards kept inside it. The first, the *Treasure Hunter*, has to find the actual chest and the other two, the *Pirate* and the *Knight* have to find a part of the 4 digit combination code which is divided into 2-digit code parts.

We developed a system with three individual game sub scenarios and game interfaces for the players to access the game world through, using various web and pervasive game technologies. Augmented reality is utilized as a means to creating more immersive user experiences; profile information is retrieved from social networks to personalize game rules; several contextual parameters are incorporated, e.g. user location and speed, time, temperature, light intensity and environment sound level. Last, the implemented game scenarios are diverse with respect to their orchestration needs, technologies used, game session duration and effort required by the players. The reason of this diversification is, firstly, to cater for different user needs and preferences and, secondly, to serve as a testbed to assess the impact of different pervasive game modes on user perception, appreciativeness and experience.

The remainder of the article is structured as follows: Section II provides a summary of our game design considerations and the main contributions of our research. Section III details the overall scenario as well as the individual sub-scenarios of *Barbarossa*. Section IV discusses implementation issues and Section V concludes the article and draws directions for future work.

II. DESIGN CONSIDERATIONS

The design and development of *Barbarossa* contributes in understanding the impact of pervasive technology usage, orchestration level, player-generated content and implicit player personal information to the overall player experience in pervasive games; the goal is to extract guidelines with regards to the above pervasive game design aspects in order to enhance player experience.

Pervasive computing technologies used in *Barbarossa* vary depending on the game session duration and the effort level required by the player. To address the lack of services availability and technological limitations reported during the evaluation trials of existing game prototypes (e.g. questionable precision in localization [4, 6, 8, 9]), *Barbarossa* supports diverse game modes: (a) prolonged game sessions which require restrained player effort and involve moderate usage of technology, and (b) short time scale game sessions, which require higher player effort with intense use of technology. That way, we assess game experiences and estimate the impact of technology usage under diverse game modes employing different design features. Finally, *Barbarossa* eases the customization of game settings according to player's characteristics, allowing her to choose the communication means with the other player and select the game mode that suits best to her gaming preferences.

The main research contributions of our pervasive game prototype are the following:

- We designed individual game modes able to function with few or no orchestration actions so that the player can play anywhere, anytime. This design approach helps in preventing user frustration associated with orchestration [8]; it also allows users to interrupt the game with uncompleted tasks and resume later on, while it does not necessitate supervision by orchestration teams.
- Many of the orchestration actions needed are supported by the players themselves by adding those actions seamlessly into the game process, in the form of player-generated game content.
- Respecting the players gaming experience and preferences, we designed individual game sub scenarios and provided various collaboration options to let players choose their own gaming style and communication method and feel comfortable throughout the game process.
- We take into account the localization and communication uncertainty that emerges when using GPS, WLAN and GPRS. Along this line, we design game modes with different localization and communication requirements, relevant to the game mode effort level (restrained/intense).

III. GAME SCENARIO

The plot of *Barbarossa* is based on the legendary adventures of the Barbarossa pirate brothers (Aruj, Ilyas and Hizir). Returning from a trading voyage to Tripoli, the boat of Aruj and Ilyas was stopped and plundered by a galley from Rhodes Island, operated by the Knights of St. John who killed Ilyas and kept Aruj as a prisoner [11].

In our generic game scenario a modern *Treasure Hunter* discovers (within an old book) some information about a chest filled by the three Barbarossa brothers with their most valuable treasures. The chest was hid by Hizir somewhere in the city of Mytilene and locked by the other two brothers with a four digit combination lock setting 2 digits of the code each; unlocking the chest would require the presence of all the three brothers. Getting obsessed with the chest and desperate to find more clues, the *Treasure Hunter* performs a black magic ritual managing to contact the spirits of the *Knight* that wants to kill Ilyas and a *Pirate* member of Hizir Barbarossa's crew who is on his way to free Aruj and commands them (on her dream) to learn about the code parts that unlock the chest, while she searches for the chest in the town of Mytilene. To fulfill the overall objective of *Barbarossa*, all three players should cooperate to trace and open a physical chest locked with a four digit combination lock.

In order to provide incentives to the players acting the three roles, *Barbarossa* defines three individual sub goals. One player who will play as the *Treasure Hunter* has to find the

real chest, locked with the four digit combination lock. The four digit combination that unlocks the chest is broken down into two parts consisting of 2 random digits of the combination code each. The *Knight* and *Pirate* players follow and complete an individual sub scenario each that rewards each one of them with one part of the code that unlocks the chest.

The player enrolled in finding the chest acts the *Treasure Hunter* and pursues the *Treasure Hunt* scenario, trying to find the chest hidden by Hizir Barbarossa in the city of Mytilene roaming into the city streets, scanning QR-Codes and consuming points (points, also called 'hints', are credited by the two other players, as discussed in Section IV.A) to receive clues that will help her in spotting the hidden chest.

The player acting the *Knight* pursues the *ManHunt* scenario, according to which Ilyas managed to survive the battle and hid in a building, somewhere in the town of Mytilene; he will try to escape there from and embark on a vessel. The *Knight* player's objective is to approach and kill Ilyas (visualized via an auto-moving map marker), chasing him along the city streets of Mytilene, so as to deter his escape.

The player acting the *Pirate* pursues the *Set Me Free* scenario. According to the later, following the battle among the Knights of St.John and the *Pirate*s, Aruj was captured by the Knights. However, their ship was damaged; hence, the Knights had to wait for another vessel to leave the island. To prevent anyone from attempting to free Aruj they decided to continuously move him around the city of Mytilene so no one could guess his exact location. They ordered two *Guards* to take turns in moving Aruj around and change shifts when needed. The *Guards* though, being passionate gamblers, started betting their shifts on rolling dices. The looser guards Aruj for the next shift and decides about the next position that Aruj will be transferred to.

Apart from the *Pirate* player, the *Set Me Free* scenario is supported by two other players acting the *Guards*. The *Guards* play a dice game taking turns in throwing two dices until someone scores a double. The looser will set Aruj position into the real world using a custom Google Maps application, while the *Pirate* will roam the city streets chasing an AR view of Aruj and his *Guard* trying to approach them, kill the *Guard* and free Aruj.

Upon the successful completion of the *ManHunt* and *Set Me Free* scenarios, the system generates a message with the 2-digit code part rewarded to each player. All enrolled players need to complete their individual game goals successfully, as those are complementary in their common mission to unlock the chest and fulfill the overall game objective. The game sub scenarios are pursued individually. The players can play anywhere, anytime. In the *Set Me Free* scenario the *Pirate* player can invite players to act the *Guards* roles in order to acquire the required part of the code.

The players are introduced into the game, receiving a full manual of *Barbarossa* and using a smartphone device each, running separate custom applications, which support the individual scenarios. Certainly, a player may use her own smartphone, having downloaded and installed the respective application.

Upon completing their scenario the players need to gather on the same physical location to unlock the chest found by the *Treasure Hunter*, using the code parts obtained by the two other players, so as to claim their rewards. Fig. 1 illustrates the *Barbarossa* player roles, the scenarios pursued and goals of each one as well as the overall goal of *Barbarossa*.

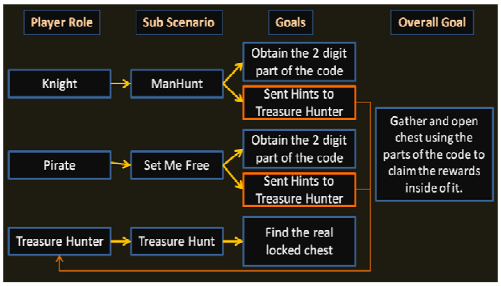

Fig. 1. The game structure of *Barbarossa*.

IV. Implementation Aspects

A. The Treasure Hunter

Recall that the *Treasure Hunter* pursues the "*Treasure Hunt*" scenario, which corresponds to prolonged time scale, requires restrained player effort, involves moderate usage of technology and requires some short of orchestration actions (i.e. QR-Codes placed at specific city locations, used to deliver clues) as well as the participation of recruited individuals along the game session.

The *Treasure Hunter* roams the cityscape scanning QR-Codes (using a custom Android application) questing the actual location of the treasure chest. For the QR-Code scanning, the ZXing barcode image processing library is used [12]. When the player scans a QR-Code, she gets a clue that leads her to the next one. The clues typically vary from a short textual or visual description to e-mail accounts of people that the player should communicate with to ask information about the next clue. Alternatively, clues may be obtained from SunSPOT sensor node devices [13] running a custom Java application: a hidden clue is revealed (visualized 'on the air' through the node's LED lights) when the player performs a specified gesture.

The player consumes energy units in order to retrieve clues. The consumption of energy units is proportional to the average environmental sound level recorded during the QR-Code scanning (the idea is that the *Treasure Hunter* should remain unnoticed while searching for the chest). To recharge energy points the *Treasure Hunter* has to switch to "Sleep" mode. The application then exits and, when resumed, it recharges one energy point for every minute it remained in 'sleep' mode.

The *Treasure Hunter* can also use hints to obtain additional information about the clue she currently receives. The hints are secretly embedded into the QR-Code scanned by the player and saved locally into her device. However, the number of the times a *Treasure Hunter* can ask for a hint varies (it depends on the effectiveness of the *Pirate* and the *Knight* players in completing their respective game goals). When the *Treasure Hunter* needs to use a hint she has to connect to the Internet and press the 'Hint' button. Then a hint relevant to the current clue is revealed and the hint points into the database are reduced by one (the number of available hints is decreased). Fig. 2 presents the main interface of the *Treasure Hunt* application.

Fig. 2. The Treasure Hunt interface showing an example clue text pointing to a church nearby the city's castle, visualizing a church picture relevant to the clue text, the remaining energy points and the sound level of the last scan.

B. The Knight

The *Knight* Player pursues the *ManHunt* scenario, which refers to a short time scale game session that requires intense player effort with increased use of technology. The *Knight* Player acts outdoors, operating a custom Android application on a device equipped with GPS receiver and 3G or Wi-Fi radio. The area around the port of Mytilene, where Ilyas will try to escape from, is visualized through a Google maps interface.

When a GPS position fix is acquired, the game is ready to begin as soon as the player presses the 'Start' button. Thereafter, a pirate marker (representing Ilyas) appears on a map interface, following a route starting at a random historical city location (his hideout) and ending at a randomly chosen point around the port of the city. The trajectory followed by the *Pirate* player is provided through a Google Map Directions API [14] invocation. At the game startup, the system retrieves the current temperature on the *Knight*'s location (via the Yahoo Weather web service) and uses it to adjust Ilyas speed. If the temperature is below 20 ^{o}C, Ilyas moves faster, resulting to a shorter game session to protect the *Knight* from exposing herself to the cold, between 20 ^{o}C and 30 ^{o}C Ilyas moves in normal speed, while above 30 ^{o}C in slow speed.

The player (*Knight*) is informed about the distance between her and Ilyas as well as between Ilyas and his escape point and has to approach Ilyas in a distance up to 20m in order to be able to shoot him (by pressing a cannon button) and claim the part of the code kept by him. . When the game ends, a message is displayed revealing the 2-digit part of the combination code that unlocks the chest.

To motivate the *Knight* to raise her effort and enhance the collaboration among players, we have incorporated a feature that allows the *Knight* Player to credit the *Treasure Hunter* with hint points by calculating, at the end of the game, the distance among the location of dead Ilyas and his intended escape point. A long distance is interpreted as higher player effort to catch him, crediting the *Treasure Hunter* with higher number of hints. Fig. 3 presents the *ManHunt* application interface.

Fig. 3. The ManHunt interface presenting the Ilyas Google Maps marker, the *Knight*'s position, and the distance between Ilyas and *Knight* as well as between Ilyas and his intended escape point.

C. The Pirate

Set Me Free! refers to a short timescale game session requiring intense player effort and utilizing a multitude of technologies including GPS, Wi-Fi/3G connection and AR. This game mode is self-orchestrated as the orchestration actions required are seamlessly embedded into the game process and undertaken by the players themselves having the form of player-generated content.

Upon the game startup, the *Pirate* (executing a custom Android application) appoints her physical location (GPS position fix) as the game's center. This is an orchestration action embedded into the game play, ensuring the game's portability (potentiality to establish the play-space anywhere). The *Pirate* can launch the AR application (powered by the Android Augmented Reality Framework [15]) and tap the 'Center' marker that appears inside her AR view to denote she is ready to play.

Having set the center, the *Guards* may enter the game. As mentioned above, the *Guards* roll two dices aiming at scoring a double. First they need to execute a custom Android application called *Shift Adapter*. The application firstly adapts the game rules based on actual *Pirate* player's age and presents them to the *Guards* (this is detailed later on in this section). It also advises them to tap/roll one virtual dice each. The looser has to guard Aruj first. Namely, she is required to set Aruj position (tap it on a Google Map interface) anywhere within a 100 meter radius from the center determined by the *Pirate* outdoors player. Fig. 4 illustrates the Swift Adapter application interface.

Fig. 4. *Set Me Free Swift Adapter* interface featuring the virtual dices, Aruj position displayed as a guarded pirate marker, the outdoors *Pirate* player last known position shown as a pirate marker and the center as a castle marker.

The Aruj AR marker (obtaining its position on the real world from the Aruj position set by the *Guards*) then appears on the outdoor *Pirate* player(s) *Set Me Free* interface; the *Pirate* should then chase Aruj and kill the guard by tapping the AR marker when she approaches him in distance up to 15m. The setting of Aruj outdoor position is also an orchestration action embedded seamlessly in the game play and game scenario. Following the initial setting of Aruj on the map, the *Guard* that lost starts rolling the dices. As soon as she scores a double the other *Guard* takes turn in setting the next position for him and Aruj and starts rolling the dices again. The goal of the game for each of the *Guards* is to prevent being the one that guards Aruj at the time that the outdoor *Pirate* approaches and kills him (i.e. resembling the 'hot potato' game). To allow the *Guards* follow some short of a Aruj relocation strategy the last known position of the outdoors *Pirate* player is revealed at the time they decide for the next Aruj position. Aruj AR marker position updates every time the *Guards* set one new position from him.

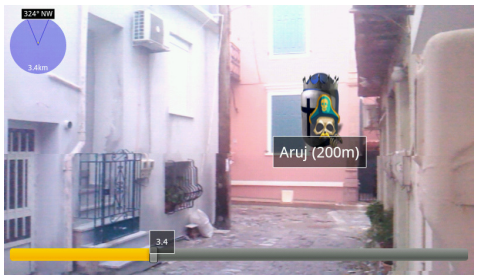

Fig. 5. Augmented Reality view of Aruj*'s* position.

The game ends when the outdoor player manages to approach and release Aruj (the *Guards* are notified about this incident). Upon releasing Aruj, the *Pirate* receives a part of the unlocking code. Also a number of hints corresponding to her average speed of the *Pirate* during the game session are transferred to the *Treasure Hunter*. Throughout the game session, the system records the average speed of the outdoor *Pirate* player and retrieves, if available, her age from her Facebook profile. The above information is stored online and used to adjust the game rules (time taken for dices rolling) in future game sessions. Fig. 5 presents the *Set Me Free* AR application interface.

V. Conclusion and Future Work

This article introduced *Barbarossa*, an outdoors role-playing pervasive game that employs open and portable design architecture. *Barbarossa* eases the relocation of the game at any urban setting, simplifies the game setup and prevents the players' frustration typically experienced due to human orchestration.

We designed intense short time scale game modes (*ManHunt* and *Set Me Free*) which function without requiring orchestrations (or seamlessly embedded into the game process), using player-generated content and enabling intense use of technology. We also designed a prolonged time scale game mode (*Treasure Hunt*) which involves moderate usage of technology and requires some orchestration actions.

Our prototype is currently undergoing a testing phase. In the future we aim to refine the game modes already implemented and develop another game scenario which will enable users to consume player-generated content. Furthermore, we plan to perform extensive user trials and formally assess the impact of pervasive technologies penetration and orchestration on the gaming experience.

References

[1] S. Benford, *et al.*, "Bridging the physical and digital in pervasive gaming," *Communications of the ACM,* vol. 48, pp. 54-57, 2005.

[2] B. K. Walther, "Atomic Actions -- Molecular Experience: Theory of Pervasive Gaming," *Computers in Entertainment,* vol. 3, pp. 1-13, 2005.

[3] I. Chatzigiannakis, *et al.*, "The "Hot Potato" Case: Challenges in Multiplayer Pervasive Games Based on AdHoc Mobile Sensor Networks and the Experimental Evaluation of a Prototype Game," *CoRR abs/1002.1099,* 2010.

[4] A. D. Cheok, *et al.*, "Capture the Flag: Mixed-Reality Social Gaming with Smart Phones," *IEEE Pervasive Computing,* vol. 5, pp. 62-63, 2006.

[5] S. Benford, *et al.*, "Uncle Roy All Around You: Implicating the City in a Location-Based Performance," *ACM Advanced Computer Entertainment,* 2004.

[6] S. Benford, *et al.*, "Can You See Me Now?," *ACM Transactions on Computer-Human Interaction,* vol. 13, 2006.

[7] C. A. Lindley, "Trans-Reality Gaming," presented at the Proceedings of the 2nd Annual International Workshop in Computer Game Design and Technology, 2004.

[8] J. Fischer, *et al.*, "Final Crossmedia Report (part II) – Epidemic Menace II Evaluation Report," 2006.

[9] I. Herbst, *et al.*, "TimeWarp: Interactive Time Travel with a Mobile Mixed Reality Game," presented at the Proceedings of the 10th International Conference on Human Computer Interaction with Mobile Devices and Services, 2008.

[10] B. Guo, *et al.*, "Design-in-play: improving the variability of indoor pervasive games," *Multimedia Tools Appl.,* vol. 59, pp. 259-277, 2012.

[11] A. Konstam, *Piracy: The Complete History*: Osprey Publishing, 2008.

[12] D. Wave. (2012, *Multi-format 1D/2D barcode image processing library with clients for Android, Java.* Available: http://code.google.com/p/zxing/

[13] SunSPOT. (2011, *Sun SPOT World Home.* Available: http://www.sunspotworld.com/

[14] G. Code. (2012, *The Google Directions API - Google Maps API Web Services — Google Developers.* Available: https://developers.google.com/maps/documentation/directions/

[15] W. J. *android-augment-reality-framework*
A framework for creating augmented reality Apps on Android. Available: http://code.google.com/p/android-augment-reality-framework/