Developing Multiplayer Pervasive Games and Networked Interactive Installations using Ad hoc Mobile Sensor Nets

Orestis Akribopoulos, Marios Logaras, Nikos Vasilakis, Panagiotis Kokkinos, Georgios Mylonas, Ioannis Chatzigiannakis, Paul Spirakis
Research Academic Computer Technology Institute (RACTI) &
Computer Engineering and Informatics Department, University of Patras, Greece
{akribopo, logaras, basilakn, kokkinop, mylonasg, ichatz, spirakis}@cti.gr

ABSTRACT

We present here Fun in Numbers (FinN, http://finn.cti.gr), a framework for developing pervasive applications and interactive installations for entertainment and educational purposes. Using ad hoc mobile wireless sensor network nodes as the enabling devices, FinN allows for the quick prototyping of applications that utilize input from multiple physical sources (sensors and other means of interfacing), by offering a set of programming templates and services, such as topology discovery, localization and synchronization, that hide the underlying complexity. We present the target application domains of FinN, along with a set of multiplayer games and interactive installations. We describe the overall architecture of our platform and discuss some key implementation issues of the application domains. Finally, we present the experience gained by deploying the applications developed with our platform.

1. INTRODUCTION - MOTIVATION

Most recent advances in microprocessor, wireless communication and sensor/actuator-technologies envision a whole new era of computing, popularly referred to as pervasive computing. Autonomous, ad-hoc networked, wirelessly communicating and *spontaneously interacting* computing devices of *small size* appearing in *great number*, and embedded into environments, appliances and objects of everyday use will deliver services adapted to the user and the time, the place, or the context of their use. Activities that, until recently, were conducted "passively" between people or between people and their environment, are now technologically invisibly augmented in order to offer new kinds of experiences. As a result, we see classrooms enhanced with visual equipment and small, portable computers (PDAs, netbooks, etc.) for the students, near field communications (NFC) technologies emerging and personalized advertisments, or even mixed-reality games and virtual worlds.

In this work, we focus on activities performed when people visit museums, archaeological sites, natural parks, etc. In these kind of activities we have already witnessed the integration of personal guided tours, enhanced reality frameworks and other technologies. The potential involvement of children's entertainment makes the design and development of such activities very challenging. They must be carefully organized, so as to be simple in their interaction, constantly keeping the attention of the participants and of course to offer to children some satisfaction or knowledge. When we refer to entertainment we do not necessarily mean the one that is provided by the various gaming platforms like the Nintendo Wii, Microsoft Xbox360, etc. We are more interested in entertainment installations, that make children get involved, make them move, require them to think, understand and react. This comes along the additional trend of detaching from traditional gaming environments, evident by the massive success of mobile platforms, like the Sony PSP and the Nintendo DS, or even mobile phones with gaming capabilities like the iPhone.

In this work, we propose a number of such entertainment installations that are based on the usage of ad hoc mobile sensor networks (WSN). Wireless sensor networking is a field well-studied both in its theoretical and more practical aspects, and we feel that such knowledge can be utilized to provide new services and products. Moreover, recent advances in ad hoc networks, the mobile phone industry and sensor technology have helped to introduce new products that integrate various kinds of sensors into mobile devices and handsets, at a reasonable cost. We argue here that the potential of combining sensors and mobile devices in order to produce new exciting entertainment applications is very large and is still terra incognita in many of its aspects. In this respect, we describe a number of such implemented applications and also propose a number of other ones.

The basic idea described here is that a user (e.g., a child visiting a museum or an art exhibition), is given a specialized mobile device through which he/she can participate in a number of networked interactive installations with a certain entertainment/educational aspect. The participation of children in an entertainment event is initiated by simply approaching the installation (e.g., the video wall, the mechanical installation, actuators and other). Also, a key characteristic of these installations is that players participate and interact with the installations by moving, running and gesturing as a means to perform entertainment related actions, using sensor devices. The use of sensors such as accelerom-

Ace 2009, Oct 29-Oct 31, 2009, Athens, Greece
© ACM 2009 ISBN: 978-1-60558-864-3/09/10...$10.00

eters, e.g., in the case of the Nintendo Wii gaming console, has already been proved a major success. Whenever a child wants to leave one installation and visit another, he/she can start a new interaction or continue a previous one. We also assume that a centralized entity can store data related to the interactions of a child and enhance his/her new interactions based on previous gained knowledge or exhibited behavior.

Such entertainment installations have a number of common characteristics that must be identified so as to point out the benefits of our proposal. Firstly, the interactions between children and the installations are based on the sensing of *presence* (enter/start or leave/pause interaction) and *movement* (perform entertainment-related actions). Both are provided using wireless sensor technologies. Also, the installations can operate with numerous participants, adding multiplayer characteristics to the installations. The participants can be easily differentiated based on the device they carry; however, no security or personal data are in any way in danger, since the children's identity is connected with the devices they carry. Children can easily exchange devices and continue playing. Moreover, a centralized control center exists for the central coordination and the management of the installations.

In order to develop, deploy, operate and administer such installations we created a unified, software and hardware-wise, framework. We utilize Sun SPOT [22] nodes as our prototype implementation hardware platform, which provide the basic functionalities of wireless sensor network nodes. Our framework is based on a hierarchy of layers for scalability and easy customization to different installations (heterogeneity). A number of services are currently implemented, common in all installations, allowing location awareness of wireless devices in indoor environments, performing sensing tasks while on the move, coordinating basic distributed operations (e.g., mutual agreement).

The remainder of the paper is organized as follows. In Section 2 we report on previous work and in the subsequent section we identify fundamental issues related to the application domain discussed in this work. In Section 3.1 we present several games implemented using our framework, which give us an insight of the requirements, while in Section 3.2 we propose a number of interactive installations targeted toward museums and other types of exhibitions. In Section 4 we describe the overall architecture of our framework. A set of experimental results are discussed in Section 5. We conclude our work in Section 6.

2. RELATED WORK

There is a large body of work regarding the pervasive games genre. The aim of the IPerG EU-funded project [10] was the investigation of the pervasive gaming experience and the implementation of a series of "showcase" pervasive games. Several other works present implementations of pervasive games, see e.g., [7, 15]. Some examples of well-known pervasive games are *CatchBob!*, *Uncle Roy All Around You*, *SupaFly* [11], *Human Pacman* [9]. In [24] several interesting issues are raised, regarding the theory around pervasive games. In [11] the authors evaluated how people perceive and play a pervasive game in normal, everyday settings. In [6] and [13] a number of issues (e.g., handle uncertainty, architecture, orchestration) related to the designing of pervasive games are considered. In general, most works focus on the design issues raised by specific games. Some of these works additionally try to generalize such issues regarding the design of pervasive games overall. In contrast, in this work we go the opposite way and build a framework for the easy and efficient creation of new pervasive games, with which developers are alleviated from the burden of handling many of these common issues.

The last couple of years we have witnessed the use of devices that utilize physical body movement input on a commercial scale, such as the Nintendo Wii. The big commercial success of this product has also led other major companies, such as Microsoft and Sony, to introduce their own versions of physical input devices. Such configurations range from using solely accelerometers and gyroscopes (Nintendo) to using a combination of such sensors along with cameras (Sony), and to using only cameras (Microsoft). Although these products allow for the detection of physical movement of the players, they differ somehow from our own platform and target application domain. We aim at applications and installations that: i) require input from multiple sources, and not only movement detection, but also (potentially) light, humidity, presence, temperature, etc., ii) are played by multiple players in non-controlled environments, so the use of cameras would potentially be non-applicable since players' views from cameras could be hindered by one another, iii) may require precise synchronization *between* the people participating (in terms of time or movement), which also may be difficult to implement otherwise, iv) overall we aim at a different application domain. Finally, our work is based on open-source platforms and tools, whereas these are closed source commercial products, both hardware and software-wise.

Until only recently, the vast majority of related work was in the context of using PDAs or smartphones as a personal guidance system, while possibly incorporating mixed-reality concepts in their operation. Our work differs substantially in that it is meant to be used in *engaging* participants in a more interactive process with both e.g., the museum exhibit and the visitors themselves. There is also a certain body of work on using pervasive methods and tools for interactive installations with a certain museum orientation aspect to their use, e.g., [21, 14, 3, 23, 5]. Such works differ from our approach by not using multimodal sensor inputs in order to provide additional interaction with the users, they follow a much more centralized architecture than ours and also provide less capable services of synchronization and situation awareness. A quite thorough discussion on such matters, regarding the integration of sensor networks in entertainment-oriented applications, is included in [8].

Regarding the visualization and generally the interfacing of pervasive and interactive installations, there is also a great deal of work revolving around Processing [20], which is a scripting language developed initially for visualization purposes but was extended toward encompassing pervasive networking and even sensor networking capabilities (e.g., [2]). We utilize Processing and its extensions as a subcomponent in our work in order to enhance the overall audiovisual experience of players/visitors.

Figure 1: Actual gameplay instances of the "Hot Potato" game played at the University of Patras campus

Although there have been some attempts to develop multiplayer games that rely on devices sensing the real world, these works are rather limited in number and scope, and are even less in a pervasive multiplayer context. Examples of WSN-based games are [18, 17]. In particular, [17] describes a concept close to our own, implemented using a mobile agents middleware, but with a narrower overall scope and without the innovative user interface used in our work. Also, in [4] a tourism-oriented locative game is presented, which uses certain simple gestures to recognize user input, quite similar to the ones we are using, with the aid of mobile phone-integrated accelerometers, whereas we are currently using WSN nodes. A platform with different purpose than ours, but in the same spirit regarding the actual hardware platform and the user interface provided is described in [16]. Our platform also considers and handles a number of WSN issues like congestion, communication disconnections and others, in a transparent to the players way. These issues have been relatively extensively investigated in theory and it is interesting to see how a real implementation handles them and of what kind are the performance issues that appear. Regarding the networking technology used in similar games, [12] describes the experiences from using Bluetooth to develop a multiplayer game. Our work focuses on using IEEE 802.15.4-compatible devices. To the best of our knowledge our work is the first, in this context, that uses these type of devices.

All of the works mentioned above differ substantially from our own approach, at least in the sense of their goals; we target mainly games that *involve multiple players, rapid physical activity, gesturing*, whereas in the majority of the existing approaches, intense physical activity is not a prerequisite. Furthermore, we use small-in-size devices that are *easy to carry and use*, whereas in the examples mentioned above PDAs or rather expensive mobile phones are used. This severely limits the ways devices are used by the players, partly because of their size and the fear of potentially damaging them during gameplay. Also, in such games some portion of storytelling is involved in the process of making and actually playing the game; we currently lean toward *less storytelling-based games*, which makes the development and set up of the games far less tedious. We think that these features will be critical for the wider acceptance of such games in the near future.

3. FUN IN NUMBERS

We present now the basic system requirements and key design goals of Fun in Numbers (FinN), a platform for developing pervasive applications and interactive installations for entertainment and educational purposes. We attempt to identify the differentiating factors of our approach from already existing ones, along with some of the respective implementation requirements. We believe that these key factors are common in both application domains that use ad hoc and mobile sensor networks for entertainment and education.

Simultaneous participation of multiple users: we envisage games and installations where groups of players participate, potentially in large numbers. The players will be in close proximity, most probably in indoor environments, and will have to engage in such applications by either interacting between themselves or with an infrastructure provided by the organizing authority. Depending on the nature of the application, players may have to cooperate or compete with each other, e.g., to reach the goals of a team-based game inside a museum, and this may be done in a real-time fashion. Regarding implementation, this assumes that there is a reliable neighborhood discovery mechanism, along with proximity detection, location-aware and context-aware providing mechanisms to the software and the players. These mechanisms are required to scale to a large number of players and to different area sizes.

Multiple types of inputs: we envisage the utilization of a plethora of inputs, the most general of which are presence, motion and other types of sensors. Such inputs will, in the majority of cases, be provided by the mobile devices carried by the participating players. In simple words, this means that e.g., pupils or museum visitors will carry mobile devices that are able to sense their location (absolute or relative to each other and specific landmarks), their movement (both in terms motion detection and gesture recognition) and other physical measures (e.g., the device could sense if the player is in a warm/cold or light/dark place). Therefore an expandable architecture is required to cover all the different sensors that can be used on a single device and be reported to the upper layers of the system, along with mechanisms for reliable motion detection and gesture recognition. In the additional case of using cameras throughout the system, respective mechanisms for the same actions must be used.

Distributed network operation: the use of embedded sensors and ad hoc networking capabilities requires that the software executed on the mobile devices carried by players is based on lightweight mechanisms. The complex parts of the system's logic need to be implemented at the fixed infrastructure. Furthermore, depending on the final application, further functionalities may be required that rely on real-time coordination and complete knowledge of the users' whereabouts, or are executed in a disconnected part of the network. It is therefore necessary for the architecture to be distributed and to involve a certain level of modularity and heterogeneity. Delay-tolerant mechanisms can be activated to ensure the correct operation of the system and/or reliable multihop or multicasting mechanisms may be necessary to cover all possibilities of communication between players and the infrastructure.

Need for synchronization and coordination between players: in most games players are competing or cooperating in order to reach/fulfill the goals set in a specific application. Players have to directly interact with each other and the overall system in a synchronized way. Such synchronization schemes should cover updates of the state of the players and the system, and also possibly coordinate the ways that the users move and act inside the playing field. Mutual exclusion, agreement and leader election mechanisms may be used to ensure the correct operation of the system.

Non-conventional interfacing methods and use of actuators/haptics: the participants should be able to decipher both their personal and/or their team's status/score while engaging in the proposed interactive schemes, and also the system interfaces should reflect the location and context awareness inherent in such situations. The use of actuators such as lights turning on/off, opening/closing doors, haptic interfaces, etc., will enable a more immersive experience.

3.1 Pilot Multiplayer Games

In order to further demonstrate the capabilities of our system we present here four pilot games that we have implemented. The key characteristic of these games is that players engage in interactions with each other and their surrounding environment by moving and gesturing, as a means to perform game-related actions. The player, as a physical entity, is the center of the game. The players' input is kept to a minimum (e.g., by means of performing a specific gesture) or is indirect (e.g., based on the location of the player). Similarly, the feedback of the game to the player is again minimum (e.g., win or lose) and some times sporadic (e.g., indicating that the player reached a specific location or is close to an opponent). There is minimum need for continuous visual feedback compared to most video games played today, e.g., through a display. FinN games are meant to be played in every place and at every time, with or without any fixed game "backbone" infrastructure. After the game is over, players can upload the data collected by their devices to a social networking web portal.

We selected these four games to reveal, to a certain extent, the variety and the joyfulness of the games produced by FinN. The *Moving Monk* and *Casanova* games feature location-aware services. The *Casanova* and *Hot Potato* demonstrate real-time player interaction. The *Anonymous* offers context-aware services. The *Hot Potato*, *Casanova* and *Anonymous* support delay tolerant networking. The *Casanova* requires input related to the players' motion and relative distance. We have implemented, tested and played these games in a number of occasions (e.g., see Figure 1) in an attempt to evaluate FinN. We remark that all these games have been very easily developed, deployed and played, due to our FinN framework and the services it offers.

Moving Monk: Each player in the game is called a "monk", moving continuously amongst a predefined set of "temples". The goal for each player is to visit all of the temples as fast as possible, perform specific "prayers" in each location. A temple is defined by the coverage range of the available infrastructure and the prayers performed are specific gestures. To help monks find the temples, clues can be given regarding the exact location of a temple, but in general the players are unaware of the temples' location. The winner of the game is the first monk who gets to visit all of the temples.

Hot Potato: In this game, each device held by players randomly generates a Hot Potato, which "explodes" after a specific amount of time, eliminating the player carrying the potato. Each player can pass the potato to one of the neighboring players, by performing a specific gesture. Thus, each player tries to pass the potato of her device to the other players, so as to avoid elimination by the exploding potato. If the player tries to avoid meeting (i.e., getting outside the range of) other players, then a new potato is generated in her device with high probability. As a result, more than one potatoes may be active simultaneously in each game. When a player who already carries a potato receives an additional one, then two potatoes merge. The winner is the player who last stands alive while all other opponents have been eliminated.

Casanova: This is a two-players-only game. One of the players is randomly selected as the "Casanova", while the other one as "Bianca". The goal of Casanova is to run away from Bianca, while Bianca must not lose Casanova from her sight, running when Casanova runs and staying still when Casanova does not move. The two players are informed for who is who and the actual game starts. Casanova tries to win Bianca, by running away from her, or by staying still suddenly. This game is based on ad-hoc networks, where the need of infrastructure is not compulsory. As a result, the Casanova game is easily played anywhere and anytime.

Anonymous: In this game there is the "Anonymous" and the "apprentices"; the Anonymous is very powerful, while the apprentices are in search of the Anonymous' identity. The goal of the Anonymous is to hide her role, locate the apprentices and eliminate each one of them by performing specific gestures. The goal of the apprentices, on the other hand, is to prolong their participation in the game by avoiding the Anonymous. If the apprentices uncover the Anonymous, they can eliminate her by combining their powers, that is, perform specific gestures simultaneously. Last player standing - or more than one if apprentices - is the winner.

3.2 Pilot Interactive Installations

We now present five pilot networked interactive installations that we are currently implementing using our platform. We expect that these schemes are simple, self explaining and should challenge players (of all ages, but mostly pupils) to interact with them. Like before, we are based on two basic sensing capabilities, presence (near a point of interest or near another person), and movement detection and recognition. A notable advantage of using mobile ad hoc sensors (e.g., Sun SPOTs instead of cameras) is that each device provides a unique identification of a player, that works similarly to an RFID and can be used for user history tracking or for enhancing the overall entertainment experience.

Clashing rocks: This installation represents an ancient ship (trireme), where players have the role of oarsmen. Each player holds an oar on which a SunSPOT is mounted, capturing and recognizing the oar's motion. While stroking, players are trying to escape from being clashed. By stroking fast and synchronized, the ship gradually gains full speed. A boatswain imposes audibly the stroking tempo, which accelerates while all players maintain their synchronization. The overall experience is enhanced by visualizing the game's progress. In this cooperative game, the basic features of synchronization, gesture recognition and visual output are combined.

Chromatize it!: This edutainment installation is based on the mixture of basic colors. The basic features demonstrated here are proximity between devices, player's input as well as visual output. A chromatic mass appears as soon as the player approaches the screen. By choosing among basic colors available on his device, the player colorizes the masses' minions. By doing so, he mixes colors, in an effort to match the color of the mass. The matching combination leads to an ever increasing difficulty of levels in chromatic complexity. More than one players can simultaneously participate.

Tug of war: In this highly competitive multiplayer game, players enter a 3D cube (approx. 2 x 2 x 2 m) on each side of which colors are floating. Each color defines a territory owned by a player. The aim of each player is to expand his territory as much as possible. This is achieved when the indicated gestures are performed properly and fast. Visual output as well as gesture recognition are the basic characteristics of this installation. Each player owns a chromatic territory, which he aims to expand over his opponents. By performing gestures faster than his opponents he manages to dominate. There is no limit to the amount of players.

Caricatura: We use the sensing and tracking history of the devices carried by users. A number of screens are placed inside a large area (e.g., inside a big museum) and each such screen is associated with a characteristic, such as e.g., a color, a piece of clothing, an exhibited item (e.g., from the archaeologic exhibits in the room). When a user approaches a screen its entity (e.g., represented with a circle) is enhanced. A simple example is the following; a student has acquired a hat and an umbrella, and interacts by changing the color of the circle, or combining colors by adding clothes to the caricatura, and so on.

Team Work: We use the sensing and the movement detection capabilities of the spots. We utilize two screens and two teams, where players going near one of the screens belong to the corresponding team. The two teams may compete in various schemes, e.g., ships that sail in the sea, cars driving in a road, a circle that goes up and down in mountains, etc. The players perform gestures so as to move the vehicles. The more players in one team and the more intense the moves the players make, the effect is amplified, e.g., a vehicle moves faster.

The Oracle: We use a number of screens inside a museum, where a number of questions are presented to the participants, with multiple choice answers about the subject of the exhibition. The basic concept is that the questions can be answered: a) by a single person making a gesture corresponding to a specific answer (e.g., a circle move for answer "A"), b) by groups of people in an election-like process. Correct answers lead players to acquire certain elements that are essential to progress to the next 'level' of the exhibition. These could be i) a single element, e.g., a key to grant access to a certain area of the museum, or ii) it could be a combination of elements from different sites - get key from room A, then get a coin from room B, and so on.

4. OVERALL ARCHITECTURE

FinN is designed and implemented targeting application scenarios where a large number of players, using wireless handheld devices with sensing capabilities, participate in various game instances and game types. These games can take place in the same or different place and time. The operation of the games may be supported by a "backbone" infrastructure that provides a number of services (e.g., localization and context awareness). The games may be coordinated by a central entity that records the games' progress. The architecture of FinN has been based upon those principles and has been implemented by a hierarchy of layers (Figure 2). Each layer is assigned a particular role in the game:

Guardian layer: This layer is composed by the devices used by players during the FinN games. The *Guardian* is the software component running in each player's wireless sensing device and uses the devices capabilities in terms of user interface, communication, etc. Protocols for the discovery and the communication with the "backbone" infrastructure and other Guardians are provided (echo protocol service). When another Guardian peer is discovered the player may be prompted for further action, by using the sensors and the buttons of her device. For monitoring the evolution of the game, each game related action is represented by an *Event*. Also, Guardian peers implement services that allow them to interact even when they are disconnected from the "backbone" infrastructure for extended periods of time. In particular, when an Event occurs, the Guardian stores it to the device memory and when communication with the infrastructure is possible, then all collected Events are forwarded (delay tolerant communication service). Also, Guardians provide a subsystem, which processes the samples of the accelerometer and recognizes gestures that correspond to game-related actions.

Game Station layer: This layer implements the "backbone" infrastructure, which is important though not necessary for all the games developed. It provides localization and

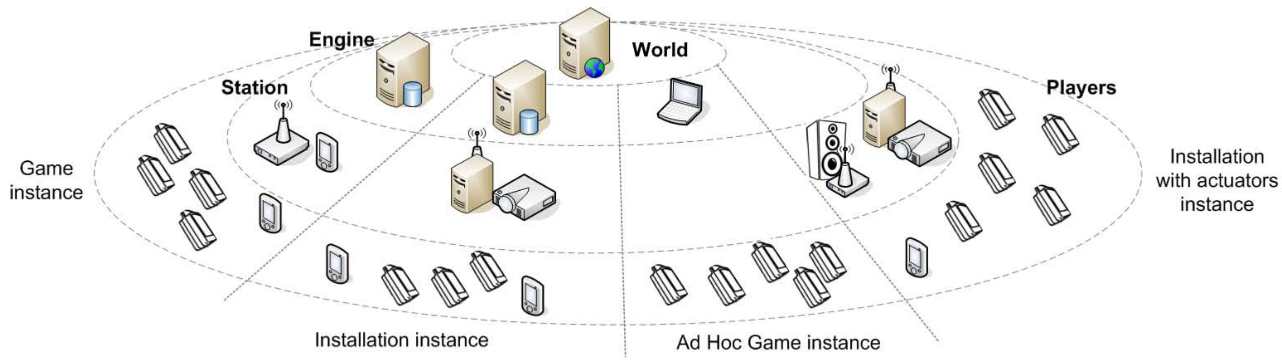

Figure 2: Overall architecture of FinN.

context awareness services and it is through this infrastructure that the data of the players are transferred to and from the higher layers of the architecture, for coordination and storing purposes. This wireless backbone is established by Station peers, with each Station controlling a specific physical area. During the initialization of each game, one Station peer becomes also the Game Engine, responsible for the coordination of the infrastructure and of the game itself. The Stations communicate with the users' devices either through local ad-hoc networks or via personal area non-IP networks and act as gateways, essentially allowing communication between the players' devices and the Game Engine. Multiple Stations can be attached to an Engine in order to maximize area coverage or the points of interest. During the initialization of a game, Stations communicate with the Engine and retrieve data such as the set of players, which are registered for this game instance, the associations between Avatars, player devices and POIs. Stations are also responsible for the Guardians initialization and for forwarding all data generated during the course of a game to the Engine.

There is also the option of using *mobile Stations* during the course of a game. In this case, such Stations operate in a slightly different way than the stationary mode - their role is primary in the context of providing location-aware services during games, while communication with the upper layers is either suppressed (by informing lower layers to use other ways of propagating game-related events to the upper layers) or carried out in a delay-tolerant mode.

Game Engine layer: Each game instance is assigned to and also coordinated by a specific *Game Engine*, i.e., it is the local authority for each physical game site. The Engine retrieves data from higher layers and stores them locally, for the duration of a specific game. In order to avoid computational and communication overhead, data between higher layers and the Engines are synchronized periodically. Thus, the processing and storage of generated events during the game is done locally. The Engine is also a control mechanism that provides game-specific services and implements various game scenarios. Communication between the Engine and the Stations is carried out through wired and/or wireless IP-based networks. Finally the Engine features an embedded Web container for providing additional game specific information to players.

World layer: The *World* layer is the topmost layer of the hierarchy, enabling the management of multiple FinN games, physical game sites and users. This layer includes the *World Portal*, which is the central point of management in the system, providing interaction with all the different game instances operating in the real world. It is also the central storing point for all game-related data, such as player-related statistics and game history. Furthermore, it allows personalization capabilities and possible interaction with external social networking sites.

5. EVALUATION

We discuss here the experience gained from implementing and operating our platform and the described pilot applications. We use Sun's SPOT platform [22], as the hardware platform for the players' devices (Guardian layer). It is a small, battery-operated device running the Squawk Java Virtual Machine, which acts as both an operating system and a software application platform, allowing programming of the devices in the Java Micro Edition (J2ME) platform. Sun's Small Programmable Objects use an ARM 9 processor operating at 180 MHz and have 512 KB of RAM and 4 MB of ROM. For radio communication SPOTs use the CC2420 Chipcon transceive (IEEE 802.15.4 compliant). They also include a simple user interface (2 buttons, 8 LEDs) and a number of sensors (accelerometer, thermistor, light). During the development and evaluation of the FinN framework, we utilized laptops and desktop computers, along with a number of Alix Gateways [1], so as to build our "backbone" infrastructure. The Alix devices carry an AMD Geode CPU at 500MHz and 256MB of RAM.The lightweight Xubuntu Linux distribution was used.

General Discussion: Our experimentation with the particular hardware setup indicates that while WSN devices can be used for playing multiplayer games and networked interactive installations, there are certain practical limitations in the implementation of such application. In particular, we investigate the performance of the system regarding the requirement of involving a large number of players.

We initially evaluated the case where players/Guardians are gathered around a Station to upload generated events. We measured the required time for all events to be inserted in the database in relation to the number of Guardians participating to the procedure. Keeping static the number of stored events on each Guardian (100 events), we increased the number of Guardians sending the events. We are particularly interested in *event reception rate* (ms/event) on the Station layer as well as *event processing rate* (ms/event) on the Station and Engine layers. The *Event reception rate* shows the time needed for an Event to be received by the Station, while *event processing rate* shows the required time for an Event to be processed on the Station and Engine layers respectively. We first conducted the experiment using one single Guardian. In this case *event reception rate* was 37.6 ms/event while *event processing rate* on the Station layer was 2.4 ms/event and on the Engine layer 5.3 ms/event. It is obvious that processing rates are significantly higher than the *event reception rate*. For this reason the Station and the Engine to remain idle for 35.2 ms and 32.3 ms per event respectively meaning that a bottleneck effect is observed on the Guardian layer. This effect is caused due to the inherent limitation of the SunSPOTs devices. This result in a pipeline effect regarding the uppermost layers of the system.

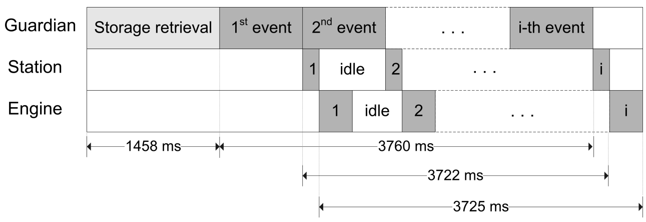

Figure 3: The pipeline effect between the three layers, Guardian, Station and Engine. One Guardian is updating infrastructure with i events in number. The durations noted are measured for $i = 100$.

Next, we increased the number of Guardians repeating the procedure described above. This creates a realistic scenario where more than one Guardians are updating the infrastructure simultaneously. We tried to decrease the *event reception rate* as low as possible, achieving a rate lower than the *event processing rate* on the Station. This would result in events to be received faster than they could be processed forcing a significant number of events pushed in the queue. We observed (Figure 4) that when the number of Guardians is increased, the *event reception rate* decreased. However, this behavior changes when a 6th Guardian is inserted in the experiment. It seems that the increased network density (6 Guardian + 1 Station) as well as the burst of data (600 events) demand a significant percentage of the already limited resources of the Guardian devices. Additionally, collisions on 802.15.4 medium lead to retransmissions resulting in a higher *event reception rate*. We conclude that if we increase the number of Guardians to more than 6 the *event reception rate* will become even higher and the network will become unstable.

Similar phenomena arise in the case when multiple players gather around a single Station. In figure 4 it is shown that the 802.15.4 interfaces of the Stations tend to become the bottleneck when a lot of players try to transmit data simultaneously; when over 6 players gather in a single neighborhood, congestion problems arise. So, in the case of installations needing real-time response to players' actions, there should be an adequate backbone network infrastructure in order to support such features. Once data reaches the Station layer, additional delays are minimal with regard to the

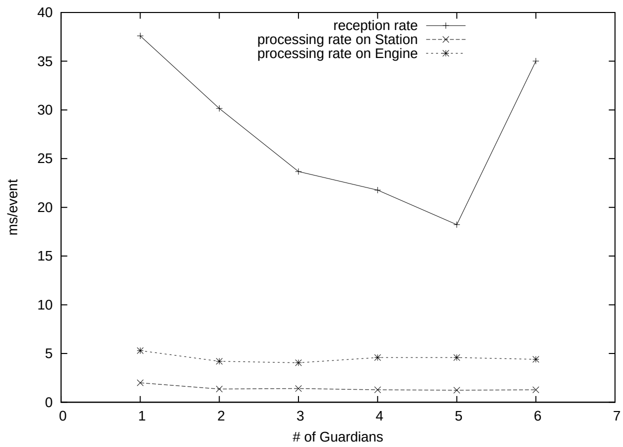

Figure 4: Event reception and processing rate on different Layers.

delays imposed by the WSN gateways. On the other hand, in games like Hot Potato, where the game is run in a disconnected mode and at the end all data is sent to the upper layer, this is a non-issue; it all depends on the type of game implemented.

Sun SPOT-related issues: We believe that SPOTs is a promising platform for experimentation and prototyping, with the ability to use Java being a really helpful feature. Still, there are a number of issues that inhibit their use in a full-fledged application, both software and hardware-wise. First of all, there are certain compatibility issues regarding the use of the Squawk VM in SPOTs and persistence in the rest of the system due to the use of different Java editions. There is also the issue of the slow flash memory installed on the devices; the experiments in figure 5 show that almost one third of the total time for sending a recorded event from the participants' mobile devices to the upper layers is due to the time needed to retrieve it from the devices' flash memory.There is also the issue of the inadequate CPU resources in order to recognize complex gestures on-the-fly.

6. CONCLUSIONS - FUTURE WORK

In this work we discussed the use of mobile ad hoc sensor networks for pervasive gaming and networked interactive installations. We also presented here a framework, called Fun in Numbers (FinN), for creating, deploying and administering multiplayer games with pervasive and locative features that employ Sun SPOT nodes as the enabling devices. We presented the architecture and implementation of our platform, along with a number of implemented games. Furthermore, we proposed a number of interactive schemes that we plan to implement in the near future. Regarding our future work, we intend to port FinN to other more powerful platforms, specifically mobile phone-oriented ones such as [19] or Android-compatible mobile handsets, in order to take advantage of the huge existing user base. These devices are equipped with accelerometers and other sensors, and also provide additional user interface possibilities. We are also interested in studying other user interface possibilities, by extending the currently used set of gestures and utilizing more advanced visualization platforms.

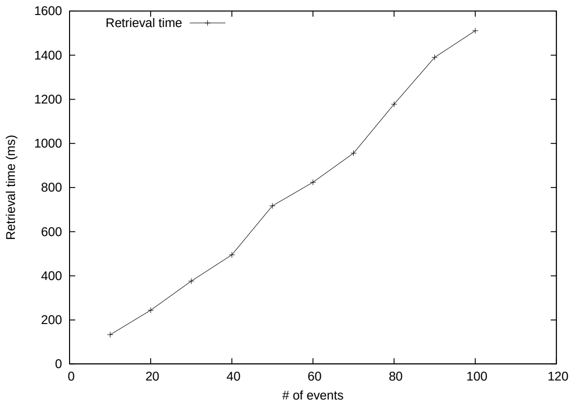

Figure 5: Stored events retrieval time.

Acknowledgements

This work has been partially supported by the European Union under contract numbers IST-2005-15964 (AEOLUS) and ICT-2008-215270 (FRONTS).

7. REFERENCES

[1] Alix Gateway, http://www.pcengines.ch/alix1d.htm.

[2] Arduino, an open-source electronics prototyping platform, http://www.arduino.cc/.

[3] T. Asano, Y. Ishibashi, S. Minezawa, and M. Fujimoto. Surveys of exhibition planners and visitors about a distributed haptic museum. In *ACE '05: Proceedings of the 2005 ACM SIGCHI International Conference on Advances in computer entertainment technology*, pages 246–249, New York, NY, USA, 2005. ACM.

[4] R. Ballagas, A. Kuntze, and S. P. Walz. Gaming tourism: Lessons from evaluating rexplorer, a pervasive game for tourists. In *6th International Conference, Pervasive 2008 Sydney, Australia*, pages 244–262, 2008.

[5] F. Barrenho, T. Rom ao, T. Martins, and N. Correia. Inauthoring environment: interfaces for creating spatial stories and gaming activities. In *ACE '06: Proceedings of the 2006 ACM SIGCHI international conference on Advances in computer entertainment technology*, page 9, New York, NY, USA, 2006. ACM.

[6] S. Benford, C. Magerkurth, and P. Ljungstrand. Bridging the physical and digital in pervasive gaming. *Commun. ACM*, 48(3):54–57, 2005.

[7] S. Björk, J. Falk, R. Hansson, and P. Ljungstr. Pirates! using the physical world as a game board. In *in Proceedings of Interact 2001*, pages 9–16, 2001.

[8] J. Burke, J. Friedman, E. Mendelowitz, H. Park, and M. B. Srivastava. Embedding expression: Pervasive computing architecture for art and entertainment. *Pervasive and Mobile Computing*, 2(1):1–36, 2006.

[9] A. D. Cheok, S. W. Fong, K. H. Goh, X. Yang, W. Liu, and F. Farzbiz. Human pacman: a sensing-based mobile entertainment system with ubiquitous computing and tangible interaction. In *NetGames '03: Proceedings of the 2nd workshop on Network and system support for games*, pages 106–117, New York, NY, USA, 2003. ACM.

[10] Integrated Project on Pervasive Gaming website, http://www.pervasive-gaming.org/.

[11] K. Jegers and M. Wiberg. Pervasive gaming in the everyday world. *Pervasive Computing, IEEE*, 5(1):78–85, Jan.-March 2006.

[12] M. Klonowski, M. Handy, and F. Golatowski. Lessons learned from developing a bluetooth multiplayer-game. In *Workshop on Gaming Applications in Pervasive Computing Environments*, 2004.

[13] H. Korhonen, H. Saarenpää, and J. Paavilainen. Pervasive mobile games — a new mindset for players and developers. In *Proceedings of the 2nd International Conference on Fun and Games*, pages 21–32, Berlin, Heidelberg, 2008. Springer-Verlag.

[14] F. Kusunoki, T. Yamaguti, T. Nishimura, K. Yatani, and M. Sugimoto. Interactive and enjoyable interface in museum. In *ACE '05: Proceedings of the 2005 ACM SIGCHI International Conference on Advances in computer entertainment technology*, pages 1–8, New York, NY, USA, 2005. ACM.

[15] I. Lindt and W. Broll. Netattack - first steps towards pervasive gaming. *ERCIM News*, (57):49–50, April 2004.

[16] D. Merrill, J. Kalanithi, and P. Maes. Siftables: towards sensor network user interfaces. In *TEI '07: Proceedings of the 1st international conference on Tangible and embedded interaction*, pages 75–78, New York, NY, USA, 2007. ACM.

[17] L. Mottola, A. L. Murphy, and G. P. Picco. Pervasive games in a mote-enabled virtual world using tuple space middleware. In *NetGames '06: Proceedings of 5th ACM SIGCOMM workshop on Network and system support for games*, pages 29–36, USA, 2006. ACM.

[18] S. N. I. Mount, E. I. Gaura, and R. M. Newman. Sensorium games: usability considerations for pervasive gaming. In *SIGDOC '05: Proceedings of the 23rd annual international conference on Design of communication*, pages 146–150, USA, 2005. ACM.

[19] OpenMoko website, http://www.openmoko.org/.

[20] Processing, an open source programming language, http://processing.org/.

[21] J. Schroyen, K. Gabriëls, K. Luyten, D. Teunkens, K. Robert, K. Coninx, E. Flerackers, and E. Manshoven. Training social learning skills by collaborative mobile gaming in museums. In *ACE '08: Proceedings of the 2008 International Conference on Advances in Computer Entertainment Technology*, pages 46–49, New York, NY, USA, 2008. ACM.

[22] Sun Spot website, http://www.sunspotworld.com/.

[23] H. Tarumi, K. Yokoo, S. Nishimoto, K. Matsubara, Y. Harada, F. Kusunoki, S. Kim, and Y. Mizukubo. Open experiments of mobile sightseeing support systems with shared virtual worlds. In *ACE '06: Proceedings of the 2006 ACM SIGCHI international conference on Advances in computer entertainment technology*, page 13, New York, NY, USA, 2006. ACM.

[24] B. K. Walther. Atomic actions – molecular experience: theory of pervasive gaming. *Comput. Entertain.*, 3(3):4–4, 2005.