RAPID PROTOTYPING OF CONTEXT-AWARE GAMES

Kimmo Koskinen (kimmo.m.koskinen@iki.fi)[1], Riku Suomela (Riku.Suomela@nokia.com)[2]

[1]*Lappeenranta University of Technology, Finland*
[2]*Nokia Research Center, Finland*

This paper presents an evaluation of a context-aware game design and implementation process. The importance of gathering and disseminating context data for context-aware applications is widely recognised. However, developing context-aware applications is a demanding task without proper architecture. The architecture for context-aware applications should also support the development process. In this paper we review rapid context-aware application prototyping through the use of MUPE platform.

Keywords: Context-aware games, Location, Prototyping, MUPE.

1. INTRODUCTION

Mobile applications should adapt to the environment in which they are used, to better support their user. Location-aware applications are already commonly used, as navigation assistants, such as TomTom (http://www.tomtom.com/) and Route66 (http://www.66.com), are widely used in cars and mobile devices. Location aware services also help in finding nearby services, such as shops or restaurants. However, apart from navigation, there are no other widespread location-aware applications available today.

The mobile device's environment has more attributes than location and this information could be further used to help the user in his or her everyday activities. Context-awareness refers to the application's ability to adapt to the changing environment (Dey, 2000), and location-awareness is a subset of context-awareness. Context-awareness opens up much more alternatives than relying purely on location.

In order to create and test new context-aware applications rapidly, there needs to be a solid platform for implementation. Toolkits, such as Context Toolkit (Salber, *et al.*, 1999) allow developers to test how context can be used in different situations. Complete application platforms, such as the Multi-User Application Platform (MUPE) (Suomela, *et al.*, 2004) make the application development very quick, and developers can concentrate on the applications, not enablers.

In this paper, we present how context-aware applications can be developed quick and easy with a complete application platform supporting context-awareness. We have implemented a plethora of context sources that can connect to MUPE, and offer developers a quick and easy way for application development. This paper is a follow-up to our rapid prototyping of location-aware games (Suomela, *et al.* 2005), where we examined the development of location-aware games in 24 hours. In this paper, we extend the work to cover context information in general, not just location-awareness.

Location is easy to understand for the developers, since it is easy to map the movement in the real world into the virtual world of the application. Location can be easily used as a direct input for character movement in the virtual world. When we consider other context information, such as weather, the mapping is still straightforward since weather can be described with a set of numerical values (speed and direction of wind, temperature, luminescence), but the application designers are faced with a more complex question: what entities and how should this information affect? In addition, from the user's perspective the weather changes very slowly, which is another great change from location-awareness.

With this research, we wanted to find out what kind of applications the developers would make, provided they can concentrate on context-aware application development. We knew from our previous experiences that location-aware applications were straightforward to implement, but when extending the focus to context-awareness in general, this might not be the case. There are many location-aware applications available, but context-aware applications are not as common. We wanted to find out what context information were used, and why, and was location still the most utilized even if there were many more available. Further, we were interested on how the developers saw the development process and the platform, and could it be improved.

We studied this setup by arranging a week long (five days) intensive course on context-aware game development. This course was offered to master-level students with hands-on intensive development. The students were given a task to develop in groups of 2-3 people a context-aware game (or application) during the course. The week consisted of lectures on game design, MUPE programming, and context-awareness. The rest of the time, the students worked in teams, first designing and then implementing their games.

This paper presents the results of the study divided into following sections. Section two presents other platforms for developing context-aware applications, and other work that is related to the paper. Section three presents the MUPE platform in principle and describes the context information sources and how the context information was aggregated. Section four describes the course teaching and development process. Section five presents a selection of the game applications created. Section six presents a review of the platform, context-awareness and game design. Section seven presents conclusions on the development of context-aware games with the MUPE platform. The final section eight presents issues in the our current study.

2. RELATED WORK

This paper continues our work started in (Suomela, et al. 2005) where we created rapid prototypes of location-aware games. In this work we look context-awareness in general, create context sources to MUPE application platform, and let developers create their own games with the platform. Several context-aware games have been developed in the past, such as Pirates! (Björk, *et al.*, 2001) and ARQuake (Thomas, *et al.*, 2002). These games use the real world as the game arena and instead of accurate context recognition; they rely on the application being fun.

There are many other tools in addition to MUPE that enable rapid development of context-aware applications. Context Toolkit (Salber, *et al*., 1999) uses context widgets to mediate the information from the source to the applications, and the developers can focus on the applications. Phidgets (Physical Widgets) (Greenberg and Fitchett, 2001) help in developing physical user interfaces. Since any physical object conforming to the API can be combined to the system, it helps greatly in the development process.

ARToolkit (http://www.hitl.washington.edu/artoolkit/) is a widely used tool for developing Augmented Reality (AR) applications AR applications are always context-aware since they are always aware of the real world around them. The system excels in video AR and applications such as the magic book have been made with it (Billinghurst, 2001). MIDAS (Yim and Nam, 2004) is a system that allows the connection of external I/O devices into a system. The system integrates the inputs in Macromedia Director allowing rapid development of applications. Director is a very good choice, since in this way the application designers have a familiar tool and no need to worry about the technology.

All these platforms and toolkits have very strong features, but MUPE excels in supporting rapid development for mobile phones, and for multiple users. MUPE can be run on virtually any modern mobile phone, yet only server side programming is needed.

Instead of having architecture-oriented approach on developing context-aware applications, some arguments have been given towards general design patterns in context-aware software development as in (Rossi, *et al.*, 2005). It is true that in the end, the problems faced with software development depend on patterns, not so much the architecture. But still, a suitable platform is needed for providing the needed context information and making the leap from collecting information to enabling a software development environment.

3. CONTEXT-AWARE PLATFORM

In this paper we use MUPE (Multi User publishing System) [10] for context-aware application development. To develop context-aware applications, we have developed several context sources that are easily available for the developers. The context data sources can be divided into three groups: location, environment and physiological. Altogether five context sources were implemented for use in the application development.

3.1 Multi-User Publishing Environment

MUPE is an application platform that allows rapid development of mobile multi-user context-aware applications. MUPE clients use J2ME, and each application server is written with J2SE. Most current and recent mobile phones support J2ME allowing MUPE to be used in hundreds of millions of devices.

Each MUPE application is written into a separate application server, and one client can connect to many servers. The server uploads the functionality to the clients enabling application developers to concentrate only on server programming. This greatly increases the application development time. There is no need for mobile phone client programming, as even the mobile UI is written to the server.

MUPE Supports context information that originates either in any of the connected clients, or anywhere in the Internet - both approaches have up and downsides. The context information in the connected clients is very personal and it is easy to get reliable information concerning all the users. Still, not all the devices have similar sensors available. For example, determining location usually requires an external GPS receiver, which everyone does not have. Context information originating from the Internet is available to everyone, but it is not always personal. Weather information received from the Internet is location dependant and if the location of a user is not known, this information is not personal at all.

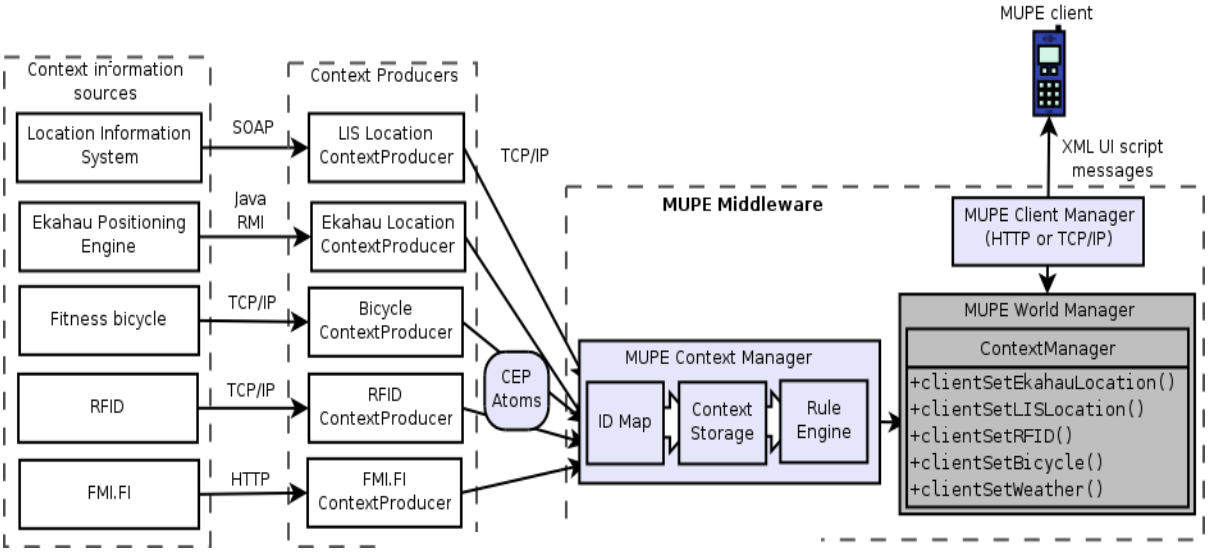

Figure 1: Context provisioning chain.

3.2 Context Information Sources

For every context data source a context producer was implemented to support the context data provisioning in MUPE. Figure 1 shows the data provisioning path starting from the originating context source and ending into the MUPE application. The basic idea is that a Context Producer is used to gather and wrap the original context data into CEP (Context Exchange Protocol) (Lakkala, 2003) format. The MUPE servers can then use the CEP data in a universal way. It is enough to write a single context producer to serve all MUPE applications. Each context producer feeds information to all MUPE applications that subscribe for the data.

The context sources can also be simulated without using real data. The MUPE environment provides a simulation framework in which one can define own CEP atoms and feed them to the MUPE servers. For example, location information can be simulated with a simple GUI that displays a map and allows a user to generate location data by clicking on the map with a mouse. Simulators of the context sources presented in figure 1 were also developed

Location information was provided by three different systems (LIS, Ekahau and RFID). The environmental and physiological context information was provided from single separate systems. An explanation of the type of data provided is presented in the following paragraphs.

3.2.1 LIS (Location Information System). LIS gathers and provides cell-based (id and description) location information from a WLAN network. An interface for making queries and tracking the location of WLAN devices or their users is provided (user can have one active device at any given time).

SOAP (Simple Object Access protocol) over HTTP was used as a communication protocol. The context producer first polls for changes in the location for a defined set of devices and then wraps the LIS location data into a CEP (Context Exchange Protocol) atom which is forwarded to the MUPE servers.

3.2.2 EPE (Ekahau Positioning Engine). EPE provides map/x,y and area based location information by using signal 'fingerprinting'. The location of a device is calculated by using a probabilistic method to compare current signal measurements to a calibrated signal fingerprint of an area.

A Java RMI based SDK was provided which allows a push -like continuous receiving of location estimates for devices. The context producer registered a set of devices for tracking through the EPE SDK and wrapped the location data into a CEP atom, which was again forwarded to the MUPE servers which used that particular context producer.

3.2.3 RFID (Radio Frequency Identification). Passive RFID tags were used together with Access 7 tag reader. The reader has a serial interface through which it can read the passive tags within a couple of centimetres.

The RFID reader was interfaced with a dedicated program that read tag sightings through a RS 232 connection. The data was relayed to a context producer program with a TCP connection. The context producer then wrapped the RFID tag id into a CEP atom and relayed the context data to MUPE servers.

3.2.4 Weather data. Environmental context information was provided from the web service of the Finnish Meteorological Institute (FMI, http://www.fmi.fi). The service provides weather measurements from different locations in Finland. The weather data contains information about temperature, humidity, wind speed and direction, air pressure and cloudiness. The measurement data is updated every hour, since weather is not changing too rapidly from a human perspective.

The operation of the weather context producer was simple: fetch the web page containing the data with HTTP, parse the information and form a CEP atom which is forwarded to the MUPE context manager.

3.2.5 Recumbent exercise bike. Physiological context data was provided from a fitness exercise bike. The exercise bike can measure the current speed of the bike and the heart rate of the user. This data was read from the bike using a serial connection (RS 232) with a dedicated program. This is a very interesting case in the context producers since this device is location-static, that is, the device is in a fixed location. In case users would use this, they would have to tie their application to locations where the bike would be.

A dedicated reader program again relayed the measurement data through a TCP connection to a context producer program. The context producer then wrapped the heart rate and bike speed into a CEP atom and forwarded this data to the MUPE servers.

A basic implementation of a MUPE server which received all the above presented context data was implemented and provided as a basis for the game implementation for the participants in the event. The placeholder project takes in the context data, but does not use it for anything. The application developers were free to decide what to do with it.

4. APPLICATION DESIGN AND IMPLEMENTATION PROCESS

To study the development cycle of context-aware games with MUPE, we arranged a week long intensive course. The course offered both lectures on game design patterns (Bjork and Holopainen, 2005) and hand-on examples to the MUPE platform and the context source arrangements. The course was divided into theory (only a game design report required, less credits) and practical part (implementation of the game and a design report, more credits). The plan was to present the software platform and the game design patterns during the start of the course and to allow the participants to design and implement their own applications. All of this was done during five days, without the participants having prior knowledge of the MUPE platform. The week was divided into following events.

Day 1: Game design patterns
Introduction to game design patterns was given as a lecture. The participants were divided into groups and they were given a task to make a draft of a game. They also received comments on the game ideas.

Day 2: Tutorial to MUPE
A hands-on tutorial session to MUPE environment was given. This was the first touch into the implementation platform for context-aware games. At the same time the groups could fit their game design ideas into the MUPE environment. Preliminary implementation of the game projects started.

Day 3: Introduction to context-awareness
Introduction to context information sources available for the game ideas was given. Background information on the context sources was given to enlighten how the positioning systems worked. Also demonstrations on how to use the context sources with the available hardware (laptops, RFID tags and reader, the recumbent bicycle) and software (MUPE context manager object methods) were given. The groups were able to start adding the context information sources into their game projects.

Day 4: Introduction to other context projects
This day consisted of a lecture of other context information projects led by VTT Technical Research Centre of Finland, e.g. (Kolari, et al, 2004). Implementation of the game ideas continues.

Day 5: Final application presentation
~~Implementation deadline for the~~ games was on the afternoon. A demo session for all games was held. A fast evaluation is done and game ideas and implementations so far are presented.

TABLE 1 - All games, and contexts they used. If game name is in Finnish, the English translation is indicated in brackets.

Game	LIS (cell location)	EPE (floor/area/ x,y)	Fitness bicycle (speed)	Fitness bicycle (heart-rate)	Weather
Aarteenetsintä (Treasure hunt)		floor/room			
Ennalta ehkäisevä isku (pre-emptive strike)		floor/room			
Agents		floor/room			
Speed-fillarointi (speed biking)			X	X	
Suunnistusseikkailu (Orienteering adventure)		x/y			
Taistelu Tieto-Sähkö -talossa (The battle of IT-Electro building)		floor/room/ x,y			X
Magic mushroom race		x,y			
Cannon Fortress					X
Spy VS Spy		floor/room			
Tykkipeli (Cannon game)					X
Drink some beer		floor/x/y			
Lumisota (snow war)		floor/room/ x,y			
Flag hunt		floor/room/ x,y			
Hunters		floor/room		X	

5. THE EVENT

90 people participated the intensive course and 38 of them did the implementation part. 14 context-aware games were developed with the MUPE environment during that time. We gathered feedback from students by using a questionnaire which contained 54 questions on the arrangements of the course. The intensiveness of the course produced positive as well as negative feedback. The MUPE environment was found to be fit for this type of rapid development and the style of development encouraged people to help each other. On the other hand the intensiveness of the course produced physical stress. Evaluation of the course in general can be found in (Porras, *et al.*, 2005).

The teams that developed the applications provided also a concept document on the game together with the functional demonstration that was presented on the last day of the event. Some of the game concepts and their use of the context information are presented here. The main context used was location context, as seen from table 1, which lists all games that were developed.

Drink some beer
Game description: Find the beer pint located in the game world. Drink the pint to raise alcohol level. When alcohol level is high enough, use it to teleport the beer pint.

Contexts: location (floor and x/y). Real world location of a player was mapped into game world location.

Snow war
Game description: Players either attack a wall with snowballs or defend it. The defending players are notified when an attack occurs and they have to run to defend the target area. The attackers use a minigame to shoot the balls. The defenders use a minigame to block the snow balls. If the ball hits the wall, the defenders have to fix it. The wall is divided into sectors which represent locations in real world (floors divided into areas). Figure 2 shows the screens from the mini games from left to right: wall status, attack, defend.

Contexts: location (floor and x/y). Used for mapping real world locations into game world.

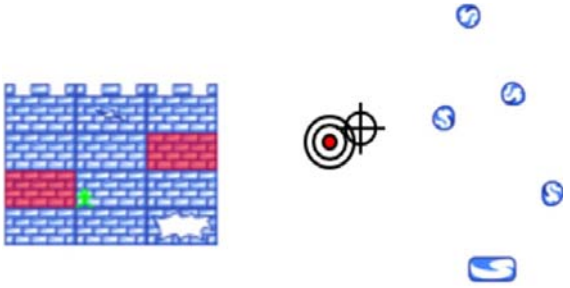

Figure 2: Snow war mini games from left to right: wall status, attack, defend

Hunters
Game description: Find other players by searching the game area. Fight them and charge fighting power with the recumbent bicycle.

Contexts: location (floor / rooms), physiological (recumbent)

Treasure hunt
Game description: Find treasures scattered around the game area. Real world locations are mapped into the game world area. Players must move in the real world to look for treasures. Moving from one location to another is limited by a wait-time; a player must choose moves carefully. Traps can be set into possible treasure locations to harm other players (increases their waiting time on a location).

Contexts: location (floor / room)

Pre-emptive strike
Game description: Players gather pieces of a bomb to blow up the base of the opposing team. The peaces are scattered around the game area. The players must move in the game area to carry the bomb pieces into the base of the opposing team. The players can try to steal the bomb pieces from the players by moving close to the player carrying the bomb. When a piece is stolen, it is randomly placed in back into the game area. When all the pieces are brought to the base of the opposing, the bomb goes off and the team who constructed the bomb wins. Figure 3 shows a screen from the game representing the state of the bombs of the two teams.

Contexts: location (floor / room). Bomb peaces and the bases of the teams were placed into the game area. The locations of the real world were mapped to the virtual world.

Figure 3: State of the two bombs Pre-emptive strike game.

Speed biking
Game description: Train with the bicycle and keep the speed given by the game. The game gives a range for the speed of the fitness bicycle which the player must meet. Also the heart-rate of the player is displayed.

Figure 4 shows two samples of the main screen from the game.

Contexts: physiological (recumbent speed and hear-rate)

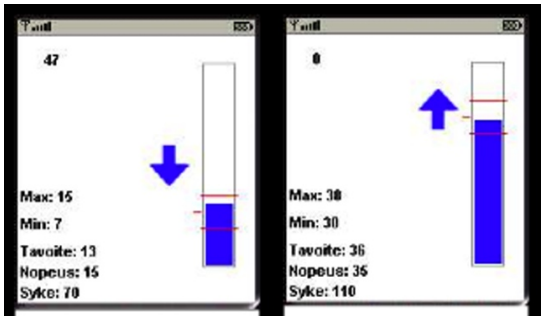

Figure 4: Main screen of speed biking. The speed meter should stay within the limits given by the game.

Magic mushroom race
Game description: Players gather mushrooms scattered around the playing area by moving using the location context information. Players see the distance to the nearest mushrooms and gather them by walking towards the mushroom. When the mushroom is gathered, a new mushroom is placed randomly on the playing area. The mushrooms are used to challenge other players. The player who puts the most mushrooms into the challenge, wins and scores points (if he started the challenge, no points are awarded of surviving the challenge).

Contexts: location (floor / x,y)

Cannon game
Game description: A variation of the scorched earth game in which the wind speed is taken from a real measurement source. Two players try to shoot each other with the cannons. The current speed and direction of wind affects the projectiles of the cannons. Figure 5 shows the main screen of the game.

Contexts: environmental (weather). The speed and direction of wind affects to the trajectory of the projectiles shot by cannons.

Figure 5: The main screen of the cannon game showing a projectile hit.

6. RESULTS FROM THE EXPERIMENT

The event provided us a lot of information on the platform, context-aware game design, and context-awareness. Main parts of the feedback are discussed here in three subsections, platform, contexts and game design.

6.1 Platform

The server part of the software was conceived as easy to learn and use but the XML scripts used in the client proved difficult to debug. Efficient use of XML scripts was not always clear, which can lead to pushing time critical logic into the server. This again slows down the applications because of the high-latency GPRS connections used between the server and the client.

MUPE documentation was perceived as being less than sufficient. Today the situation is better with a wider developer community. Still it was considered that the platform met the needs of rapid prototyping: it was fast to build basic functionality.

6.2 Context-awareness review

The context information was brought to the application domain code with methods in one concrete object. One comment was that it would have been better to provide an interface with methods or an abstract class with template methods to unify the functionality in the applications. Also the context data could have been provided as java objects using Java collection style storage. This, however, was one consideration for the setup. How easily the students could use context information, and how would they like to use it. The setup chosen was obviously a wrong one, and better choices will be considered in the future versions.

The context source usage is presented in table 2. Usage of the context information sources was strongly biased toward location. One reason for this can be the relative completeness of the EPE location information compared to other context sources. The location information, especially from EPE, was readily usable and provided a good "input" for the games as well as strong connection to the entities in the games (players). The good quality of the EPE source probably the reason why LIS location producer was never used. Some additional contexts, such as the temperature of the users device itself (e.g. cell phone) was desired but was not available.

TABLE 2 - Context usage distribution in the games developed.

Context Source	Usage (%)
LIS (cell location)	0
EPE (floor/room/x,y)	78.6
Fitness bicycle (speed)	21.4
Fitness bicycle (heart-rate)	7.1
Weather	21.4
RFID	0

This raises an important issue regarding context-aware application development. All developers wanted personal context information, and regarded context to be centralized on the device that is used. To better answer the needs of developers, context information originating from the network should be tied to each user of the system, when possible. In this experiment, the developers could get the weather information by tying the current location of the user to an area covered by the weather service, but this was not done. There could have been too much work associated with this, or there was no need to get the weather information in the first place.

6.3 Game design

Many of the games relied on location, and player movement in the real world. This is a very straightforward way to implement context-aware games, but it requires players to move a lot in the real world. Also, the games were designed to be fairly intensive. Most games emphasized speed - the quickest to attack, collect, locate etc. would win the game. This results in games, which last a short period of time, and are fairly physical in nature.

Four applications use two context sources, and all the rest use only one context source. This indicates that the developers chose highly focused designs, in order to finish their development process in time. The concise schedule of the code camp sets limitations to the design process, which is not always such a bad thing. It is better to test context-aware features in several different finished prototypes than to merely review the concepts.

7. CONCLUSIONS

This paper analyzed the usage of different types of context information in games. We presented a framework for developing context-aware applications, which was tested in a course held for master level students. There were three main types of context information: location (cell, floor, room, x/y), environmental (weather) and physiological (fitness bicycle speed and heart-rate). We tested the development in a five day course during which the participants developed 14 context-aware games in small groups.

With the experiment, we wanted to see what kind of games the developers would create with a solid background, and how the different context information was used in the game applications. Further, we wanted to see how the platform could be improved. Even with many other context sources, the location was by far the most widely used in the games, since more than three quarters of the applications used location. Although the location producer was technically superior to other sources, this was not the cause since the applications were first designed, and only later the context sources were added. This suggests that location is very intuitive and easy to understand when designing games. In the future, the other context sources require more preparation, in order to make them appealing. The technical quality of the platform was not seen optimal. Documentation and XML scripts were found to be the hardest parts in the development. To tackle these issues, platform development tools are being made.

Based on our experiments, the context-aware application designers should pay close attention to other context sources than location. Since it is intuitive to align the real and virtual worlds of a user based on location, this can be taken for granted, and the designers should focus on how to use other context information to improve the applications. Further, the games designed and implemented in the experiment had fairly short play times, and they required intensive concentration on the game. This might not be the optimal style of play for context-aware games. Context-aware games should react to a changing user environment, and the longer the game is played, the more the environment changes. Longer lasting games could offer interesting new opportunities for the potential players.

8. ONGOING AND FUTURE WORK

Since these code camps, many applications relying on social proximity have been made with MUPE. Social

proximity is easily obtainable with Bluetooth, since the Bluetooth radios have a unique id. Such games, offer another way to perceive the world, since the location is not directly important, but rather the people around you. Apart from location and social proximity, other contexts are not so easily perceived, nor do they change easily with user actions - walking ten meters do not change the climate.

The change, and its frequency, directly affects the way a context can be used. Games that last a long time will see many relevant context changes even from slowly changing sources. This leads to a proposition, that the game length directly affects what context sources should be used. Further, design guidelines for selecting context sources for a game whose length is known can be set. This requires further study of the context sources, and the frequency of their change.

REFERENCES

Billinghurst, M., H. Kato, and I. Poupyrev. *The magicbook: Moving seamlessly between reality and virtuality*. IEEE Computer Graphics and Application, May 2001, **Vol. 21**, No. 3, pp. 2-4.

Björk, S., J. Falk, R. Hansson, and P. Ljungstrand. Pirates! - using the physical world as a game board. In Proceedings of the Human- Computer Interaction INTERACT'01, pages 423–430, 2001.

Björk S. and J. Holopainen, 2005, *Patterns In Game Design*. Charles River Media. ISBN 1584503548.

Dey, A.K., 2000, *Providing Architectural Support for Building Context-Aware Applications*, Ph.D. thesis, College of Computing, Georgia Institute of Technology, Available from: http://www.cc.gatech.edu/fce/ctk/pubs/dey-thesis.pdf

Greenberg, S. and C. Fitchett. *Phidgets: easy development of physical interfaces through physical widgets.* In UIST '01: Proceedings of the 14th annual ACM symposium on User interface software and technology, pages 209–218, New York, NY, USA, 2001. ACM Press.

Kolari, J., T. Laakko, T. Hiltunen, V. Innonen, M. Kulju, R. Suihkonen, S. Toivonen and T. Virtanen, *Context-Aware Services for Mobile Users. Technology and User Experiences*, ISBN 951-6396-4, VTT Information Technology

Lakkala, H., *Context Exchange Protocol Specification*, 2003, Available online at: http://www.mupe.net

Porras, J., J. Ikonen, K. Heikkinen, K. Koskinen, L. Ikonen, *Better programming skills through Code Camp approach*, 16th EAEEIE Annual Conference on Innovation in Education for Electrical and Information Engineering, Lappeenranta 6-8.6.2005, ISBN 952-214-052-X

Rossi, G., S. Gordillo, F. Lyardet. *Design Patterns for Context-Aware Adaptation*. Workshop on Context Aware adaptation and Personalization for the Mobile Internet. IEEE, Workshop Proceedings of SAINT 2005. pp. 170-173, IEEE Press, New York, NY, 2005. ISBN ISBN 0-7695-2263-7.

Salber, D., A. K. Dey, and G. D. Abowd. *The context toolkit: aiding the development of context-enabled applications*. In CHI '99: Proceedings of the SIGCHI conference on Human factors in computing systems, pages 434–441. ACM Press, 1999.

Suomela, R., E. Räsänen, A. Koivisto, J. Mattila. *Open-Source Game Development with the Multi-User Publishing Environment (MUPE) Application Platform.* Proceedings of the Third International Conference on Entertainment Computing 2004 (Ed. M. Rauterberg), 308-320, Lecture Notes in Computer Science 3166 Springer 2004.

Suomela, R., K. Koskinen, K. Heikkinen. *Rapid Prototyping of Location- Based Games with the Multi-User publishing Environment Application Platform*. Proceedings of The IEE International Workshop on Intelligent Environments, June 2005, 143-151.

Thomas, B., B. Close, J. Donoghue, J. Squires, P. D. Bondi, and W. Piekarski. *First person indoor/outdoor augmented reality pplication: Arquake*. Personal Ubiquitous Computing, 6(1):75–86, 2002.

Yim, J.-D. and T.-J. Nam. *Developing tangible interaction and augmented reality in director*. In CHI '04: CHI '04 extended abstracts on Human factors in computing systems, pages 1541–1541, New York, NY, USA, 2004. ACM Press.