TAMPEREEN TEKNILLINEN YLIOPISTO
TAMPERE UNIVERSITY OF TECHNOLOGY
Julkaisu 601 • Publication 601

Riku Suomela

Constructing and Examining Location-Based Applications and Their User Interfaces by Applying Rapid Software Development and Structural Analysis

Tampere 2006

Tampereen teknillinen yliopisto. Julkaisu 601
Tampere University of Technology. Publication 601

Riku Suomela

Constructing and Examining Location-Based Applications and Their User Interfaces by Applying Rapid Software Development and Structural Analysis

Thesis for the degree of Doctor of Technology to be presented with due permission for public examination and criticism in Tietotalo Building, Auditorium TB111, at Tampere University of Technology, on the 14th of June 2006, at 12 noon.

Tampereen teknillinen yliopisto - Tampere University of Technology
Tampere 2006

ISBN 952-15-1600-3 (printed)
ISBN 952-15-1832-4 (PDF)
ISSN 1459-2045

Abstract

Location-based applications exist at the border of the real and virtual worlds. These two realities are merged into a single experience that provides some added value for the user. Objects in the real world, whose position can be determined, can be projected to the virtual world, and with proper technology, the virtual data can be projected onto the real world.

Personal location-based applications serve the needs of an end user and are easily available for them. An interface for personal location-based applications is preferably carried by its user all the time, and possible devices include mobile phones and wearable computers. These devices are very different in nature, and they allow different kinds of applications and user interfaces.

This Dr. Tech. dissertation deals with location-based applications and their development. The dissertation is composed of eight research papers, which deal with the construction of location-based applications, user interfaces, and associated theory. Three research methods have been used in this dissertation. Constructive research methods have been applied in location-based application and platform development, empirical methods are used in application evaluation, and theoretical analysis on location-based application UIs.

In this dissertation we propose new user interface techniques, and a method for analysing the user interfaces of location-based applications. Rapid development of multi-user location-aware applications have also been studied, with emphasis being on user generated content. Finally, a new way to analyse the structure of location-based applications is proposed.

Preface

Writing this dissertation was started right before my first child was born, and it is about to be defended after my second was born. Although I managed to get two children while only finishing one dissertation, I do not consider this such a bad feat. I definitely do not consider finishing one dissertation to be twice as hard as raising children. Still, it feels great to be done with the dissertation.

I have many people to thank, who helped and encouraged me during the work. My supervisors Dr. Juha Lehikoinen and prof. Tommi Mikkonen were the biggest reason I got the dissertation finished. They gave great comments and helped to shape this dissertation. My external examiners prof. Mark Billinghurst and prof. Kimmo Raatikainen gave great comments on how to improve my work. Because of your help, I finished this.

There are many other people who were irreplaceable for the completion of the dissertation. My co-authors and co-workers in the papers: Kimmo Roimela, Ilkka Salminen, Eero Räsänen, Jouka Mattila, Timo Koskinen, Ari Koivisto, Kimmo Koskinen and Kari Heikkinen. In addition, Tero Hakala, Jussi Holopainen, Juha Kaario, Elina Koivisto, Harri Lakkala, and Timo Nummenmaa all helped me with this work. Well-eaters gourmet club kept me well fed physically and mentally throughout the work. It is a privilege to spend time with all of these people.

Our working environment at Nokia Research Center (NRC) Tampere has been one great motivator for me. The working environment has been stimulating, and the good atmosphere at work has boosted this work considerably. I like to thank all my group members at NRC. A special thanks goes to everyone who have been involved in MUPE development. The community has shaped MUPE, and consequently this work.

Finally, I wish to thank all my family. My parents and my siblings have always encouraged me in my life. Most of all, I wish to thank my wife Katariina, and my children Ilari and Isla. You are my biggest motivation.

List of Included Publications

(i) R. Suomela, J. Lehikoinen. Taxonomy for visualizing location-based information. *Virtual Reality, vol. 6, no. 2*, 71-82. ©2004 Springer-Verlag.

(ii) R. Suomela, J. Lehikoinen. Context Compass. *Proceedings of the Fourth International Symposium on Wearable Computers* , 147-154. ©2000 IEEE Computer Society.

(iii) J. Lehikoinen, R. Suomela. WalkMap Developing an Augmented Reality Map Application for Wearable Computers. *Virtual Reality, vol. 6, no. 1* , 33 - 44. ©2002 Springer-Verlag.

(iv) R. Suomela, K. Roimela, J. Lehikoinen. The evolution of perspective view in WalkMap. *Personal and Ubiquitous Computing, vol. 7, no. 5*, 249-262, 2003. ©2002 Springer-Verlag.

(v) R. Suomela, J. Lehikoinen, I. Salminen. A system for evaluating augmented reality user interfaces in wearable computers. *Proceedings of the Fifth International Symposium on Wearable Computers (ISWC), 2001, 77-84. ©2001 IEEE Computer Society.*

(vi) R. Suomela, J. Mattila, E. Räsänen, T. Koskinen. Augmented Reality for a Casual User: Designing Tools for Interaction with the Virtual World. Proceedings of the Computer Games and Digital Cultures Conference *, 185-190. ©2002 Tampere University Press.*

(vii) R. Suomela, 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. ©Springer.*

(viii) R. Suomela, 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.* ©*IEE.*

The permissions of the copyright holders of original publications to publish them in this thesis are hereby acknowledged.

Contents

Chapter 1

Introduction

Every human is familiar with the reality they live in; it is natural to act and interact with the real world. The real world is bound to well known rules, and people are familiar with it. In addition to the real reality, we may have many realities in the digital domain, which are referred to as Virtual Reality (VR). The real and virtual worlds are separate, but with proper technology the two can be merged into a single experience that combines them in a meaningful way. This dissertation combines the real and virtual worlds using location, and analyses the characteristics of location-based applications and presents a series of solutions for their construction.

VR is an artificial environment that is supposed to feel like a real environment to a user. This feeling can be achieved in many ways, such as a 3D environment with rich interaction capabilities, or by creating an immersive multi-user environment for participants with rich story immersion, such as Multi-User Dimensions (MUD). MUDs were early multi-user environments in the Internet that allowed the participants to take an active part in the virtual worlds. A pure virtual world is intended to replace the real world, but the real and virtual worlds can also be combined into one experience.

Augmented Reality (AR) combines the real and virtual worlds by overlaying interactive digital information on top of the real world. There are several technologies enabling AR, and several ways to build AR applications. One approach uses a Head-Mounted Display (HMD) to overlay the data on top of the real world, whereas another uses small hand-held devices as a window to the virtual data.

Mobile Augmented Reality (MAR) refers to AR systems that can be used on the move. MAR systems should be mobile, low on power consumption, and offer an easy interface for the computing services in the environment. MAR has many challenges, which mainly relate to the power consumption, as the sensors needed for AR systems must be accurate. Recently, however, a potential platform for MAR has arisen in the form of mobile communication devices such as mobile phones.

The mobile phones allow people to stay connected all the time. People can call each other whenever, no matter where they are. Further, the devices have sufficient computing power to perform many additional tasks as well such as applications and games. The devices can run stand-alone or networked applications, and monitor the status of the user, to name but a few examples. These extensions to the devices allow many interesting applications and services to be built, such as MAR applications. The mobile communication devices are in the user's pocket most of the time, making them special compared to desktop computers. The desktop computers are static, i.e. they are almost always in a fixed location, whereas a mobile device moves with its owner. This is the fundamental difference between these devices, and thus the situation of use is a very important value characterising the mobile user and device. The situation could be exploited to improve the use of a device, if relevant information was presented in a meaningful way. In a wider perspective, the devices need to be aware of their context.

Applications capability to react to the users changing situation is called context-awareness. In this work, we use Dey's definition of context [24]: *"Context is any information that can be used to characterize the situation of an entity. An entity is a person, place, or object that is considered relevant to the interaction between a user and an application, including the user and the applications themselves."* In other words, relevant dynamic information related to an object is context information and it can be used in context-aware applications.

Context-awareness allows applications and services to improve their performance or usability, with added information on the user's surroundings. For example, the device may observe the current brightness with a sensor, and increase the backlight when needed, or whenever a cheaper or faster network connection is available, it can be utilized.

One important area where context can be applied is multi-user applications. Context information can be used to connect people and users who share the same use context. For example, people who are in the same location can be grouped together in a location-based application.

1.1 Location-based applications

Location of a device or user can be used in many ways in mobile applications. Location-Based Applications (LBA) have information that is tied to real world locations. Virtual object is one term for the location-based information, and landmark is another term that is commonly used. A virtual object is often referred to as being an object that only exists in a virtual environment. In this work, we use virtual objects to refer to objects that can have a physical counterpart in the real world, or they exist only in the virtual world. Virtual objects act as a bridge between the two realities, and they are the points in which the real and virtual worlds connect. Landmark is another widely used term for the location-based objects. The difference between the the terms virtual object and landmarks is where they exist primarily. A virtual object is always at least in the digital UI, whereas objects in the real world, such as "the Eiffel tower", are referred to as landmarks.

Location-based services (LBS) and applications are already a modest business, as certain applications such as map services are making their way to mobile devices (see, for example, Mobile MapQuest [125]). Benford further points out in the tech report *Future Location-Based Experiences*, how the location-based experiences are on the verge of becoming ubiquitous [12]. There are technological problems associated with location-based applications, but the technology is in place and as mobile phones have such a wide penetration, it is up to the location-based applications to make themselves widely used.

Location-based applications can be built in many different ways. For this dissertation, we have defined four building blocks that are essential in building location-based applications:

1. *Positioning system* is the basis for everything. The position or location of a device, user or object needs to be known to build location-based applications, and there are many possible technologies to choose from.

2. *Location-based data* is the collection of objects or landmarks that is used in the application. This data can be personal, or shared among many users, and it can be stored locally, or in a network server.

3. *The application logic* determines how the data is used. A location-based application can be a map service, a game, or a friend finder. The application logic can reside in a single device only, or it can be distributed.

4. *The Human-Computer Interaction* defines the User Interface (UI) and how the users access the systems. Since location-based applications cover a very wide range of applications, there is a lot of variance in the UIs. AR applications overlay data directly on top of reality, whereas Short Message Service (SMS) based services only present a list of local objects.

These four parts are the core of a location-based application. This division resembles the layered software architecture that is widely used in communications. The ISO (International Standards Organization) OSI (Open Systems Interconnect) defines a well-known seven layered standard for networking [120]. In the layered architecture, data moves from one layer to another with well defined interfaces, and there can be both hardware and software layers.

For the analysis of location-based applications it is beneficial to use these four building blocks, as these building blocks characterise the main parts of location-based applications. Each building block has many options, which change the characteristics of the application dramatically.

First Building Block. The position of the device can be determined with a plethora of devices, and the infrastructure supporting the location-based data distribution has also many alternatives. There are two main methods to determine the position of a device:

Infrastructure assisted. In this method, the device itself does not directly locate itself, but rather an external infrastructure determines the position or helps in the determination. Examples of this situation are the mobile phone network positioning methods, such as the Enhanced Observed Time Difference (E-OTD) [2]. With this method, the device does not know its position without the help of the infrastructure.

The device determines. In this method, the device itself determines its position without the need for external parties. An example case uses Global Positioning System (GPS) for positioning [2].

Essentially, the two methods ask either "who is here" or "where am I". The main difference between these two are that if a device uses the infrastructure assisted method, it does not know its location at all if the infrastructure is not present. For example, a device utilising the Ekahau Wireless Local Area Network (WLAN) for positioning does not know anything about its location without the presence of the network [107]. When the device determines its own position (for example GPS), it knows the position without the need for an external infrastructure. One can argue that even GPS requires an external infrastructure, as the satellites in the orbit provide the signals for the GPS receivers. The difference, however, is in the network connection. A device with GPS requires no two-way communications with an external infrastructure, as it can do the signal measurements itself and determine the position. Assisted GPS (A-GPS) is another GPS technique that uses external infrastructure to improve the position of the device, see for example [27]. In A-GPS, an external assistance server does some calculations in addition to the GPS receiver to improve the accuracy of GPS. This technology makes A-GPS infrastructure assisted locationing method. The biggest problem with GPS technologies is the coverage, they usually do not work indoors too well, although there are techniques to overcome this, such as dead reckoning, see for example [56]. The technique aims at estimating the current location based on speed and direction. A relative estimate of current location based on last known absolute location can be made.

Second Building Block. Virtual objects represent the location-based data in the environment. Virtual objects are information that is tied to some real world location, such as a Longitude, Latitude, Altitude coordinate (or an area marked with coordinates) expressed in WGS84 coordinate system [30], or to a street address, to name but two examples. The data can be stored anywhere, that is, locally in the user's terminal or remotely in the network either in a central location or in a distributed manner in many locations. If the data is stored in the user's terminal, the data can be well protected and personal in nature. All data in the network is easily distributed, and sharing with many people is easy.

Third Building Block. Application logic defines how the application behaves. The virtual objects can be the same across all applications, but the behaviour of an application determines whether the user is trying to navigate to a destination point or playing a location-based game. Different applications can be built using the same positioning technology and location-based data. For example, one application could be a map application, another a tourist guide, and the third a location-based game and each application has different application logic while reusing the same positioning system and shared virtual objects.

Fourth Building Block. The user interface provides the means for the user to interact with the system. There are many approaches for interfacing with the location-based data. Some require very much processing power, whereas others can be implemented with very little effort. The input and output capabilities of the devices vary, so this part should be customisable to many different Input/Output (I/O) systems. Again, building on top of the lower levels presented here, the same position, location-based data, and application logics can be used, while the UI is changed in different applications. An application which presents location-based objects to the user can have an AR interface using an HMD, or a hand-held UI that presents the object on a 2D surface.

1.2 About this dissertation

In this doctoral dissertation we study the construction of location-based applications and focus is on application development. In addition, building the infrastructure supporting the development of such applications, UIs, rapid application development, usability experiments, and theoretical analysis of LBA UIs are studied.

The research has been carried out with two different platforms: A wearable computer and a mobile phone. On the one hand, the platforms are very similar. Both platforms are personal, as they are always carried by their users, and both platforms are meant to help their owner. On the other hand, the platforms are very different. A wearable computer is very powerful in terms of computational capabilities, networking speed and I/O, whereas the mobile phone offers less capabilities in all of these respects. Location-based applications are built differently for different platforms, although the basis remains the same.

The application requirements set constraints on the physical design of mobile device that runs the location-based application. If applications need to be self contained, all building blocks must be present in the same terminal, as is the case with our wearable computer. Also, a complex UI requires good I/O capabilities, which can be easily provided with a wearable computer. If all building blocks do not need to be on the mobile device, less and less hardware is required. Also, if a subset of a large set of landmark data is enough for an application, even a device with little memory is sufficient in many cases. Our research on mobile phones concentrates on simple UIs, that only use a subset of the complete location-based data, and the device did not determined its location with the help of the infrastructure.

The big picture of the dissertation is provided in Figure 1.1. Location-based applications are composed of four main parts, and there are many possibilities in constructing them. Each part can be either in a single device, or distributed into many devices. Single user systems can be built without any need for a network connection. We have developed techniques in each section seen in this figure. Our applications use both terminal- and network-based positioning. Landmarks are stored in the device in some applications, and distributed into the network in others. Application logic is standalone and distributed, and finally the UI is always in the user's device, but sometimes it is also distributed to other devices.

Figure 1.1 shows that there are many alternatives in creating location-based applications. Many mobile digital GPS devices have all the features in a single terminal, that is, all components are on the left-hand column in Figure 1.1. An example device is the Garmin eTrex Vista C that has a color display, GPS, and map functionality [115]. Maps can be downloaded from a PC, as the device with a USB connection. This product was designed for outdoor use and to help in navigations. The Figure 1.1 shows many more alternatives, and depending on the application at hand, there are many possibilities to choose from.

	Technology	
	Terminal	Networked
UI/Interaction		
Application Logic		
Landmarks		
Positioning		

Figure 1.1: The big picture of the Thesis.

1.3 Research questions

The research problems addressed in this work are related to user interfaces, infrastructure and application development of location-based applications. The research questions addressed in this work are:

1. How to create and analyse the user interface for a location-based application?
2. How to rapidly develop multi-user location-based services for mobile phones, and what kind of infrastructure is needed for this?
3. How to support user-creation of mobile location-based content?
4. What are the characteristics of location-based applications?

The main focus of the thesis, that is, constructing and examining location-based applications and their user interfaces by applying rapid software development and structural analysis, is well supported with these questions. As the focus is on applications and their UIs, we do not focus on the positioning technology, but rather use different technologies already available. The UI of the applications can be a complex 3D model of the environment, or a list of nearby objects, and as these examples are so radically different, this is an essential part of any location-based application. Location-based applications themselves are already on the market, and we focus on how to rapidly develop new ones to complement the existing ones. Location-based data is highly dynamic, since a lot of data is user generated, like the user location. In addition, mobile users can create data on the move, and this produces challenges for the location-based systems. The final questions is a very basic one - what are the necesary, and essential components of any location-based application, and what are the possibilities to choose from.

This dissertation studies the location-based applications, and different software and interaction techniques required in them. The work is done in two tracks: First, a high-end approach is taken with a wearable computer and second, an approach with standard mobile phone as the platform is used. This dissertation presents a series of applications that each focus on one critical aspect of location-based applications. The aspects are user interaction, infrastructure, multi-user support, and user-created content.

1.4 Research methods

The results presented in this dissertation are based on iterative design. The work started with a wearable computer, and the applications created with this were evaluated, and new applications or extensions were created. The wearable computer aspect also spawned the need for a low-end approach, that is, the mobile phone as the end-user terminal. Constructive research methods have been applied in developing the applications and infrastructure, empirical methods in the application evaluation, and theoretical analysis on location-based application UIs.

In one of the included papers (i), we present a taxonomy for visualizing location-based information. This paper forms the backbone for the UI research, as all the applications that we have constructed can be mapped to the taxonomy. We have developed AR applications for accessing location-based information with different UIs. A HMD UI with minimal screen real estate requirements allows the user to be less distracted by the UI (ii), whereas a digital map overlaid on top of the reality gives the user a good idea of the surroundings but at the same time might block the view (iii,iv). One paper (v) studies usability experiments using wearable computers, and the problems associated with them.

The location-based data available to the users could also be user generated, which is studied in one paper (vi). User generated data allows people to modify the location-based application experience with their own contributions. Multi-user aspects is another important area in location-based systems, and developing a multi-user application is also addressed in one of our research papers (vii), which also presents a platform for mobile multi-user context-aware applications. The difficulty in developing location-based applications is addressed in yet another paper (viii), which provides a platform for rapid prototyping of location-based applications.

1.5 Structure of the dissertation

This dissertation is composed of a number of research papers studying the above problems and a literature survey of the area. In this dissertation, we use two platforms, a wearable computer and a mobile phone. The wearable computer used in our research is difficult to make and maintain, but a well chosen mobile phone can be used without hardware modifications, and it is used by millions of people.

The rest of this introductory part of the thesis is structured as follows. Chapter 2 introduces context-awareness, and the research in the area. This chapter covers the first two building blocks of location-based applications (positioning and landmarks), and the chapter also includes a literature review in the area. Chapter 3 looks at Mixed Reality (MR) interfaces, which are essential for LBAs. This chapter deals with the third and fourth building blocks of location-based applications (the application logic and UI), and the chapter also presents a literature survey in related work. After this, Chapter 4 shows how context-awareness and MR interfaces together create LBAs. Chapter 5 introduces our research papers included in this thesis, and Chapter 6 concludes the dissertation.

This summary acts as an independent introduction to the 8 included research papers.

Chapter 2

Context- and Location-Awareness

2.1 Introduction

The real and virtual worlds are separate entities, but their merging opens up many new possibilities. The real world has items that could be controlled via the virtual world, and vice versa. Also, some objects might only exist in the virtual world, but they could be presented in the real world whenever a suitable link exists between the two. Some information needs to map the objects in the virtual world into the real world.

The real and virtual worlds can be merged in many different ways. In any case the virtual world contains data, and a suitable interface for this needs to be built. Virtual objects, i.e. objects that exist in the virtual world, can be shown to the user based on certain criteria, such as location or proximity. For example, a virtual object presenting the history of a building can be shown to the user when he or she enters the building. Such triggering criteria can be used to generate context-aware applications.

Context-Awareness refers to the application's ability to react to the changing use conditions. Dey defined context as information that characterises the situation of an entity [24]. Most information that changes in the user's environment can be used to adapt the behaviour of the applications. Dey further defines context-awaress as follows: *"A system is context-aware if it uses context to provide relevant information and/or services to the user, where relevancy depends on the user's task."* With context-awareness, it is possible to combine the real world and virtual worlds for the benefit of a user. A user might be interested in information about his or her surroundings, the status of himself or herself, or to gain access to services relating to the current situation. All this is achievable with context-aware applications, which aim at helping the end-user with the current situation, or information relating to it.

The first part of my dissertation looks at context-awareness. Context-awareness is the basis of location-based applications, and location has been one of the key subjects of context-awareness research from the beginning. However, context-awareness is not the main focus of this dissertation, so it is studied concisely.

2.2 Research on context-awareness

The research of context-awareness starts from the ubiquitous computing (ubicomp), which was introduced by Mark Weiser [103]. Ubicomp means that the environment and people are augmented with computing power and devices. Together they interact and provide services for the user whenever and wherever needed. This vision has been further refined and studied since then, and several areas of research address these same issues.

As an example, let us consider Active badge, which was a very early demonstration of a context-aware application [101]. The system was location-aware, tracking the wearers of active badges in the office environment. Services were provided for locating the wearer of the badges, and for call forwarding to the locations of the wearers. PARCTAB was another early system supporting novel context-aware applications in the areas of information access, communication, computer supported collaboration, remote control, and local operation [102]. These early applications were the starting point for the area of context-aware applications. Since then, context-awareness has been applied in many different areas, and some key areas are presented in this section.

A good starting point for context-awareness is analysing what is necessary information about the user's current situation. Abowd and Mynatt present the "five W's" that form a good minimal set of necessary contexts: *Who, What, Where, When, and Why* [4]. The list is just the starting point, and there is a lot more to context than these.

The definitions of context-awareness and context cover a wide area of information, and it is imperative to create a certain categorisation of the areas of context-awareness. Project ePerSpace has contributed the following categorisation of context information [41]:

- Environmental context characterise the environment the user and the device are in (e.g. light, humidity).

- Personal context is related to the user, that is, it defines the status of the user (e.g. pulse, retinal pattern, mood, stress).

- Task context relates to all tasks that are ongoing (e.g. explicit tasks, events).
- Social context is the user's relation to other humans (e.g. buddy-list).
- Spatio-temporal context defines the user's status in space and time (e.g. location, time, speed).
- Device context is the status of the context-aware device used by the user (i.e. device capabilities).
- Service context is related to specific services that are or are not available to the user right now (i.e. service specific information, e.g. name).
- Access context characterises the current network connection (i.e. networking characteristics, e.g. traffic performance).

These categories are very different in nature. For example, the personal context describes the physical characteristics of a user, whereas access context tells what is the connection status of the user's device. These attributes can be used to combine the real and virtual worlds. For instance, personal context can trigger pulse monitoring objects, or the user can be informed when friends are nearby. The spatio-temporal context is the most important context regarding this dissertation.

Context-aware applications require several steps before they can be built. The process is composed of gathering raw data, analysing the data to generate context information, and provisioning the context data to applications. First, the context information needs to be received from a context source. This is usually done with sensors that gather raw sensor data. Second, the raw data needs to be processed to generate relevant context information, and the context information needs to be represented with a format suitable for the distribution of it. Context fusion is needed when several context producers are used, and the context information from multiple sources in some cases need to be combined. Third, the context information is provisioned to the applications. The process of the context information retrieval and analysis is seen in Figure 2.1.

Sensor information gathering

The first step of the process seen in Figure 2.1 is to collect the context information. This can be done in two different ways. The user can have sensors that monitor the situation, or the information can come from some external provider, which in turn might use sensors. These two methods have two very

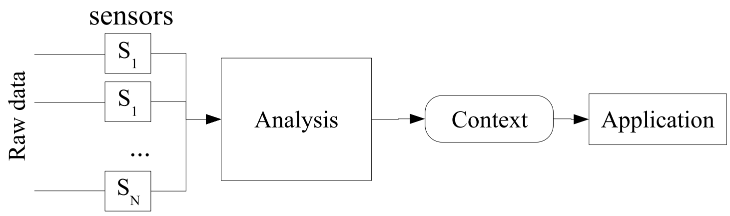

Figure 2.1: The process of gathering, processing and provisioning of context data.

different characteristics. Personal sensors usually monitor the status of a single user [32, 38, 50], (ii), whereas external context information providers monitor many different users [101, 107], (viii). Second, if the environment monitors the user, the application requires a link to the environment in order to gain the context information, whereas a personal approach allows the user to work independently, as the sensors are in the same device that uses the information.

The personal approach is more mobile, as the user can go anywhere, and the sensors are always with him or her. The drawback is that the sensors need to be carried by the user, and the analysis of the sensor data needs to take place in the user's mobile device, increasing the computational requirements. According to Moore's law, the silicon size decreases and the computational power of the devices will increase so these issues will be less dramatic in the future, if the law still holds [70]. This also applies to sensors, which decrease in size as technologies mature. Nevertheless, one can assume that the state of the art sensors are always a bit bulky and large and are not fully suitable for mobile use.

The other approach, where sensing is carried out by the environment, has the situation reversed. The user cannot be always sure that he or she is in the range of the sensing capabilities, but at the same time, the user does not need to carry any extra sensors with him or her, nor do they need to worry about the computational complexity. Some technologies, such as mobile phone positioning Enhanced Observed Time Diffenrence (E-OTD) work from far [2], although the accuracy can be lower.

There are other approaches that would mix these two, for example having sensors in the user's device, and sending this data to the environment for analysis, such as the Ekahau positioning system [107]. Also, the environment can contain the sensors, and send them to the user's device for analysis. We consider these two to be the case of tight coupling with the environment, as again the device cannot really function standalone.

The difference between the two approaches is that the user is either carrying a sensing bubble with them that monitors an area around the user, or the user is entering different sensing bubbles at appropriate times. The ubicomp vision stated that the people are surrounded by computational resources and they are in tight interaction together. A more personal approach has the user carrying most sensing, and the area of Wearable Computing is studying context-awareness from this perspective. The wearable sensor badge and sensor jacket is an example of sensor's carried by the user [31].

In location-based applications this part deals with the positioning sensor. A thorough list of locationing systems for ubicomp was written by Hightower et al., which charts all relevant locationing systems at the time [42]. Since this paper, there has be some improvements in the area. Many locationing systems are becoming cheaper and easier to setup. The audio locationing work by Scott et al. introduces a locationing system using audio that is very cheap to setup in an office environment [87].

Generating context information from sensor data

At the lowest level, the context information is just sensor measurements. These sensor measurements as such are not useful to many applications, and higher level contexts need to be derived from these. Higher level contexts can further form meaningful hierarchies that make them easier to handle. The second phase in context-awareness is analysing the raw context data, and producing relevant context information that can be provided to the applications [68]. This process of generating context information is the pattern recognition approach that is basically composed of four parts [73]: sensing, feature extraction, modeling, and inference. This is a more accurate model of the one we presented in Figure 2.1, which essentially bundles feature axtraction, modeling, and inference into a single block called analysis.

Context information can be interpreted from any sensor data, such as with computer vision algorithms out of video stream, or by simple database queries based on GPS position data. Hull also addressed these issues in his research, which he called *Situated Computing* [44]. He defined how the applications are interested in a higher level of abstraction and not in raw sensor data. For instance, it is easier to query for temperature than interpreting raw sensor data. These issues raise the need for ontologies to help the use of the information. An ontology is a formal vocabulary describing objects and the relations between other similar objects.

The context data needs to be encoded in a standard format. Many formats have been proposed, (e.g. [20, 52, 54]). A context ontology helps in systematic handling of the context information [51, 104]. When the context information is represented with some kind of hierarchy, it is easier to develop context-aware applications.

There can be several sources of the same context information that need to be combined for application specific purposes [38]. For example, GPS works best outdoors, and indoors a complementary technology should be used, such as RADAR, which is a Radio Frequency (RF)-based positioning system [7]. Thomas et al. presented a system that uses GPS and vision-based optical tracking for location [94]. Dead reckoning techniques can also assist GPS indoors, when the satellite signals are low. Dead reckoning tries to estimate the location when the primary sensor is inoperable, with for example counting the number of steps walked in a specific direction.

This merging is called sensor fusion, and it generates additional requirements for context-aware applications, and also for the analysis part. In many cases the applications are not interested in where the information came from and they want the analysis part to generate the best possible bet on the current context. In yet another case, the applications wants the information provided by all of the context producers, and the applications themselves want to make the decision based on many sources.

Context information provisioning to applications

After the context information has been generated, it needs to be sent to the applications. There are two possible approaches. Either the sensors are located in the same device as the applications, as in [52] (ii), or the sensors are in the environment [107] (viii). The first situation requires operations inside a single computing device, whereas the latter situation requires a connection to the environment, from which the data is retrieved (viii), and in this case the computing process is distributed. To enable context provisioning efficiently, a solid architecture is required that supports the task at hand.

There are many architectures for context-aware applications. One of the earliest was presented by Schilit [85]. His system architecture was intended for context-aware mobile computing systems, and the system adapts according to the location, nearby people and objects. The system is based on a dynamic environment interface, and it represents features of the mobile computing environment as dynamic environment objects. These objects are containers that store information, and they may have information about the real world, for instance about people and printers, or they can represent non-physical objects. Schilit also describes an experimental system utilising Xerox PARCTAB.

Since the early systems, there has been many different platforms and architectures available for context-aware systems. Hohl et al. presented Nexus, which was an early work designing a platform for spatially-aware applications [43]. The system divides the real world into areas and maintains models of these, and allows the users to access and interact with these spatial areas. Di Flora et al. presented a context-aware application infrastructure that is capable of providing uniform context abstractions [36]. The work presents a modular service infrastructure that can support mobile context-aware applications, by giving new context providers an easy interface to the system. The infrastructure is divided into two layers of internal context (context within the mobile device and application context) and external context sources (e.g. weather). Java Context Awareness Framework (JCAF) uses the Java platform to implement one such system [9]. JCAF has a network of cooperating context services, a robust architecture and it uses an expressive Java programming model. Another system of note is the Plan 9 from Bell Labs [8]. This system uses the file system to represent context, and all context information is presented as files. A traditional well-known system can be applied to new areas with some creative thinking.

MEssage Exchanger (MEX) was used in the wearable computing part of this dissertation, and it is a distributed architecture allowing context information coming from any source [58]. The second architecture used in our papers is Multi-User Publishing Environment (MUPE) (vii), which is a client-server architecture supporting sensors both in the mobile phone and in the environment in a ubiquitous manner.

As a summary, there are many ways in which context-aware applications can be built, but they all require a solid software infrastructure to support the development. There are many challenges left, and Raatikainen et al. list key research areas concerning the middleware of pervasive applications [77]. There are already many good platforms that have allowed many interesting applications to be made, which are looked into next.

2.3 Context- and location-aware domains and typical applications

The final phase in context-awareness are the actual applications and the application logic. Context information can improve the life of the end-user by improving usability of applications, or providing services to the user. As Dey stated, a context-aware application provides its user with relevant information and/or services [24]. This definition can be applied to almost any application, and there are numerous areas that use context information. Some interesting research includes memory augmentation [26, 81], Augmented Reality interfaces [47], (ii), healthcare [73], and social software [28, 74]. This section looks at the different domains of research that deal with context-awareness in one way or the other.

Wearable computing (e.g. [91]) is one of the main platforms, where context-awareness is studied. We have seen two kinds of approaches: First, the user is tightly coupled with the computation capabilities in the environment to create the context-aware applications [102], or the user contains all the necessary resources himself or herself [52]. The former approach is the Ubicomp approach, and the latter is used in wearable computing.

Wearable computers are worn as clothes and by nature they are the most personal of all computers. A wearable computer is always worn by its user giving an easy access to the computing resources wherever and whenever. Wearable computer can be a powerful device, as it can be larger in size than a PDA. Steve Mann has stated the essential criteria for a wearable computer [65]:

1. Eudaemonic criterion. The system is a seamless part of the user.
2. Existential criterion. The focus of the control is within the user's domain and the system behaves as an extension to the user's mind and body.
3. Ephemeral criterion. The system is always active when worn and the interactional and operational delays are non-existent.

Wearable computers have many desirable features, and also many drawbacks. The wearable computer does not need to be tightly coupled with the environment nor is it dependent on the computing resources, and thus the computer and application can be taken anywhere. Furthermore, the wearable computer is always accessible, allowing the user to access information regardless of time or space. On the other hand, as all sensing and computation take place in the wearable computer, the hardware requirements become quite high. This makes wearable computers sometimes cumbersome to wear and use, and wearable computers have not yet been widely used by the general public. It can be argued though that mobile phones are wearable computers, but this is a matter of definition.

In addition to context-awareness, research on wearable computing also focuses on the user input devices, and the distribution of the wearable devices across the wearers body. Wearable computing has spawned many innovative UI devices and interaction techniques, as seen in N-Fingers wearable finger input device [59], Interactive augmented reality techniques for construction at a distance of 3D geometry [76], and hand-tarcking for AR [89].

Memory augmentation is an area that provides the user with more information on the relevant context. A user might have encountered the objects before, and a computing device can store a lot of information that a user does not actively remember. Presenting this information to the user at appropriate situations could be a useful application.

One memory augmentation system was the wearable remembrance agent, which constantly provides information about relevant context, such as nearby people [81]. Another system, the conference assistant, also addressed these issues [26]. Memory Glasses, a more recent example, implements a subliminal visual cuing system for a wearable computer [23].

Navigation, and guiding systems are another important area in context-awareness. One example of a mobile personal guide is the Cyberguide by Abowd et al. [3, 62]. The first Cyberguide prototype was designed for indoor use. The user is shown a map of the building he or she is currently in: the map can be panned and zoomed. The You-Are-Here (YAH) symbol is shown on the map as well. If so desired, the map can be scrolled automatically as the user moves to keep the YAH symbol on the visible area. This version uses infrared tags for positioning. Another important aspect of map is whether it is forward-up or north-up aligned. North-up means that north is always on the top side of the view in the display, whereas foward up places the walking direction at the top of the display. These issues have been discussed in (iii).

Touring Machine is an AR system, which helps a walking user in an urban environment [32]. The system helps the user by augmenting data on top of the reality, and the system supports multiple display and interaction techniques. Contrary to the earlier systems, this system supports large outdoor spaces, which is always a challenging task.

In addition to Touring Machine, there are other highly innovative UI solutions for helping the user navigate in large outdoor environments. The AR systems can take a lot of screen real estate, so other solutions might allow the user to see the real world more freely. Map-In-The-Hat presents a navigation application, which uses minimal screen real estate [96]. The work focuses on navigation and guiding people from point A to B. Waypoints help the user to reach the final destination, and the waypoints are shown on the user's see-through HMD along with a compass.

Our research also explored the possibility of using minimal screen real estate in an HMD when navigating the real world (ii), [60]. Context compass adds a generic set of virtual objects on a compass that takes only a little screen space on an HMD. This approach allows a walking user to walk in the environment and concentrate mostly on the real world.

Yet another tourist guide is Smart Sight, which assists the tourist especially in foreign countries [105]. It provides speech and image recognition, natural language processing capabilities, and navigation assistance. The system is a wearable computer, where the prototype setup has two computers, one in the backpack to take care of the processor intensive tasks, and another on the waist belt.

GUIDE is an intelligent electronic tour guide using a hand-held device with networking capabilities [21]. GUIDE is targeted at tourists, and it allows navigation with the help of a map, and information retrieval, which is also context-sensitive. Google Earth is a global 3D navigator of earth, which has gained a large number of users [117]. This system provides an easy interface to geographic data.

The guidance systems are already mature and ready for a wide audience. MapQuest Mobile is one example of a system usable in mobile phones and PDA's [125]. Benefon has had phones designed for navigation on the market for a long time [113].

Creating location-based data on the move gives the users a chance to create their own content into the virtual object database. Virtual objects in many cases are a collection of data provided by a service provider. For example, a navigation service might need a Point Of Interest (POI) database to help the user in locating interesting places. Such a database contains the viewpoint of the service provider to the location, and the interesting objects in it. One database does not fulfil every user's needs. People need and want to personalize services for their own needs, and user created content enriches many applications considerably.

Virtual objects created on the move have been addressed by many researchers. Pascoe discussed the manipulation and creation of virtual objects while the creator is in motion [71]. He created the Field assistant prototype, a set of tools that help a fieldworker while observing the environment. Its Stickepad application creates Stick-e notes, which contain the GPS position, and the application runs on a 3Com PalmPilot. The notes are created from a template to speed up the creation process. Templates are very useful when working with devices that have limited input capabilities. Other research includes the Augment-able Reality that allows the user to add voices and photographs to physical objects or locations dynamically [80].

When moving from notes to 3D AR, the task becomes a lot more complex. Creating 3D objects on the move is studied in [76]. This work allows the user to create 3D virtual objects into the real world with wearable AR system. The task is highly complex, as the new 3D objects need to be created and combined with the real world in real time. This work highlights how the mobile user can contribute objects into the AR experience.

Our research (vi, viii) studies how the user's can create location-based data on the move. (vi) Studies the end-user requirements, and (viii) presents an application, in which the players in a game create the virtual world locations by exploring the areas in the real world.

Supporting technicians and fieldworkers is another area of context-aware applications. Fieldworkers are typically performing tasks that may need some kind of assistance. Knowing the object that needs repair can be used to access relevant information. NETMAN is a system supporting mobile fieldworkers collaboration and cooperation [11, 53]. The system adapts to context and presents the fieldworker with the necessary tools for the task. A more recent example is presented by Lampe et al. [55]. In this work, a ubiquitous computing environment helps an aircraft maintenance worker.

Games have some interesting properties making them suitable for integrating new technologies, which also includes context-awareness. Games are very forgiving if some context information is interpreted falsely, as a game does not need to model the reality accurately. If game content is designed to seamlessly integrate with the real world, as in AR applications, the same requirements apply. The two worlds need to be accurately aligned to create a good user experience. On the other hand, if the game world is a fantasy world on top of reality, the two do not need to be as well aligned. In this case, the virtual world is an alternative reality, which by definition is not similar to reality, or its rules.

Pirates! is a good example of a game that does not require accurate context information [17]. In Pirates!, the players moved in the real world and found new locations in the game world. The game world did not recreate reality, rather, it observed the players distance to other players and to hotspots. The key thing was to be close enough to another entity, which triggered events in the game world. This demonstrates why games are a good area to introduce new technologies; a good game design can be created with inaccurate context information.

Treasure hunting games, such as Geocaching have been around for a long time [108]. Treasure hunting games are one of the first multi-user context-aware applications available. Treasure hunting games leave objects at a certain location (or context) and the players need to find these treasures.

Treasure hunting games are fairly simple, and there are much more complex games with richer content and interaction available. ARQuake transforms the well-known Quake game by ID software into Augmented Reality, and it requires accurate alignement of real and virtual worlds, as is the case with AR [94, 95]. AR games will have a few hurdles to pass before they become affordable to an everyday user. An article by Avery et al. explores the current situation, and problems associated with AR gaming most notably registration [5]. The paper also points out how to design AR games so that they can ignore of the technological problems of AR, mainly to utilise the fact that people cannot accurately judge distances beyond 30 metres.

BotFighters is a multiplayer game played with a mobile phone, and it can be played with inaccurate position information [110]. Another similar game, Mogi, is a location-based game where players explore the real world collecting items and trading them with other players [111]. Mogi is a commercial product in Japan, and Botfighters has been released in Europe. Moreover, on the theory side, Sotamaa has written an article analysing how to use the real world as the game arena [90].

"Can You See Me Now?" is a chase game where online players were chased on a city map by actual runners on the streets [35]. All participants shared a map: online players could see a local map, whereas the runners on the streets had a global view to the map. The players used a web interface, whereas the runners had a Compaq iPAQ. These games mix the real and virtual worlds in new ways providing the players with a unique and new approach to gameplay. Uncle Roy All Around You is a continuation of this work that mixes gaming and theatre [34]. Players move on the streets and collaborate with actors to achieve the goal of meeting Uncle Roy. These games create new ways in which mobile and stationary online players can collaborate.

Healthcare and context-awareness are a good match and there are many applications in this domain. Sensors attached to a wearers body can monitor a person all the time, and thus provide a much more up to date knowledge of health, than other non automatic follow-up methods. A persons health can be monitored in real time and the status can be transmitted to a hospital, or a service for in-depth analysis. Bardram has written an article discussing applications utilising context-awareness in a hospital environment [10]. The paper discusses several scenarios for such an environment, and proposes design principles for context-awareness in medical work.

The wearable computing is a natural starting point for such personal monitoring, and the MIT wearable project MIThril has presented applications also in the area of health monitoring [22]. With such a platform that is potentially always worn, long-term healthcare monitoring is possible. Healthwear presents one such system, and it also discusses how such monitoring is needed as the population is aging and less and less nurses are available [73].

Social context is related to other humans in your current context. Keeping track of existing friends, and finding new friends is an appealing application to many people. Social serendipity proposes a system where people have profiles at a central server, and as people are close to each other, their profiles are matched according to certain criteria [28]. When an interesting person is nearby such a system could propose that they should meet. Social context, and how to quantify social signals, is further studied in [74].

2.4 Categories of context-aware applications

So far, this chapter has charted the area of context-awareness. The area is large, and the process of creating context-aware applications is composed of many steps. An early context-aware application categorization divides the applications into three different areas, as seen below [25]:

1. *Presenting information and services.* The user gains access to information in the digital domain, such as location-based data or local devices such as printers.

2. *Automatic execution of a service.* The user device is automatically adapted in a specific context, or any other automatic functionality.

3. *Tagging of information with context for later retrieval.* This means that an object is tied with context information.

The three areas: *presenting*, *execution*, and *tagging* are the main areas for context-aware applications. In our research we have applied techniques to all of these areas in different devices. We have also studied the effects of these in multi-user environments. The main target of our research has been in giving the user access to the virtual objects in a certain context. Tagging, that is, *creating* virtual objects into the digital domain has been thought of throughout our research. The automatic execution of services has been the least focused in our research, but there is a lot of interesting research done in this area, for example context-aware content adaptation. Kolari et al. introduce a system that adapts WEB and WAP content for different end-user terminals [48].

Dey and Abowd provide a list of context-aware applications in each of the three different context areas [25]. As seen in the previous section there are many more applications that could be mapped into a similar list, and this has been done in for example [67]. In addition to this categorisation, there are many other ways in which the applications can be categorised. Dey and Abowd further categorise the applications, based on the context type, which include four different components [25]: Activity, Identity, Location, and Time. Recently there have been much more detailed categorisations, such as the one used in Project ePerSpace [41]. Also, the used devices are mobile phones, wearable computers and PDAs, and the device could also provide the basis for the categorisation.

These application areas chart the area of context-aware research. These define at a high level how the different applications behave, and what are their main characteristics. The application areas are constantly evolving, as new types of context-aware applications emerge. For the different kinds of application areas, there are different platforms that enable the development of applications. These platforms are looked into next.

2.5 Context-aware platforms

Context-aware application development requires a solid platform, and there are many platforms that speed up the development. These platforms have a slightly different focus, as their main focus is on the different parts of the wide area. Each try to hide some complexity of application development, whether it being sensors, UI development or other. Next, we will present some platforms that have been presented for context-aware application development.

Context Toolkit is a system that allows easy creation of context-aware applications by introducing context widgets [24, 83]. The context widgets mediate the context information between applications and the source, and the developers do not need to concentrate on the context information.

Phidgets [39], or physical widgets, is another technology that can be used in developing context-aware applications rapidly. Widgets aim to make physical user interfaces as easy to use as widgets in graphical UIs.

ARToolKit is one of the most widely used AR application development toolkits [112]. ARToolKit uses video tracking libraries that calculate the user's viewpoint relative to markers. This approach makes the system very versatile, because video cameras are becoming ubiquitous as they reside in most mobile phones. ARToolKit has been used succesfully in a very wide set of AR applications, such as the MagicBook [16], and real world teleconferencing [13]. ARToolKit has also been ported to handheld devices, such as the pocketPC platform [100].

Tinmith is another AR platform for developing mobile AR applications [124]. The platform uses ARToolKit for the input devices, as the users hand is tracked with ARToolKit [89]. For the user viewpoint, a different approach compared to the ARToolKit is used. GPS is used for location, and sensors to determine the users viewpoint. This makes the system usable in a wider area, since no markers are needed in the environment.

Designer's Augmented Reality toolkit (DART) is an AR system that tries to take the development of AR applications further away from an engineering approach, and closer to the application designer [64]. DART is built on top of Macromedia Director, which is a multimedia development environment widely accepted by the indstry. DART integrates AR into macromedia allowing rapid development and testing of AR applications. The designers can make an AR implementation already at an early design phase. The DART system tries to move away from the low level programming required by the other toolkits, such as ARToolKit, Tinmith, and Studierstube [86]. These systems are rather hard to program from a designer's point of view, and DART sets the goal of easing the development.

Another similar system is the Media Interaction Design Authoring System [106] (MIDAS) which allows the connection of external input/output devices to the system. The system can be used to rapidly develop AR application in Macromedia Director, and the possibility to connect any external I/O device makes it very capable for rapid prototyping of context-aware applications.

Multi-User Publishing Environment (MUPE) application platform developed by us is a platform for rapid development of multi-user context-aware aplications (viii). The platform is developed for mobile phones, and it supports context-awareness generated in mobile phones, and in any other external context producer. The platform enables rapid development of multi-user applications and games (viii). Many users in context-aware applications have some interesting features. The context information of a single user affects all other users of the system and there is potentially a lot of information exchange. MUPE is a client-server architecture based system that updates the status of users in the server. Moreover, the server has grouped the users so that it knows who need the context information updates.

2.6 Evaluating context-aware applications

Context-aware applications can be evaluated in many different ways. The application concepts can be evaluated with real end-users by conducting interviews at a very early phase. This helps in setting requirements for the applications before the development starts. After the applications have been developed, the actual methods can be tested in laboratory. Laboratory testing offers a controlled environment, where the variables affecting the testing can be kept to a minimum. Laboratory testing is good for testing the usability of an application, and certain techniques in an application. The best form of testing for context-aware applications is field trials. In field trials the applications are tested in their actual use environment, which is crucial for context-aware applications, since they are meant to react to varying use situations.

Kolari et al. [48] present a large study on context-aware services. In total, the project used 98 users in the evaluations. This is a very large study on context-awareness and context-aware services. The paper also uses a plehotra of methods in the evaluations. Kaasinen [46] discusses the importance of field testing. She argues that traditional laboratory evaluations were not so useful, and in many cases the testing needed to be done in the field to satisfy certain context conditions, such as getting GPS signal. She also discusses the difficulties found in the field testing relating to uncontrolled environment, and the difficulties in technical failures.

There are methods to keep the field trials in control of the organisers. In (v) the test subjects were able to walk in the real environment, and all the infrastructure was carried by the test organisers in the test setting. This setup, however, is not generalizable for more than one users, and for many applications as it is. In this setup the test laboratory is essentially carried by the test organisers which is not an easy task.

2.7 Summary

This chapter presented the area of context-awareness, and how context-aware applications are built. Context-awareness requires sensor measurements, measurement analysis, and information provisioning to applications. Context-awareness is a very wide research area that covers areas like social, spatio-temporal, service, access, etc. From the beginning of context-aware research, location has been the key context, as seen in the research review in this chapter.

In this dissertation we have developed two different context-aware application platforms. The first is WalkMap (iii, iv), which is used in a wearable computer. The second is the Multi-User Publishing Environment (MUPE) (vii) that is an application platform for multi-user context-aware applications in mobile phones. Both platforms enable context-awareness in a wide range of devices. WalkMap can be run on PCs, while MUPE clients run on any mobile phone supporting Java 2 Micro Edition (J2ME) Mobile Information Device Profile (MIDP) 2.0.

WalkMap uses MEX, which is a small footprint, dynamic, distributed architecture in a wearable computer [58]. MEX is very personal in nature, although different wearable computers with MEX can be combined together. MEX was mainly used in a single wearable computer and in single user applications. The wearable computer in our research was equipped with a digital compass, a GPS receiver, a digital camera, and other sensors on the wearable computer creating a standalone context- and location-aware platform, which could also be connected to external infrastructure.

MUPE, on the other hand, was designed for multiple users, and for context information coming from any external context producer (viii), or from any connected user terminal. A key part of MUPE relating to context-awareness is context manager that handles any external context information coming to the system. A single external context producer can feed information to any number of MUPE applications. The external context producers need to encode their data in a standard format, and MUPE uses the Context Exchange Protocol (CEP) [54]. An inference engine can create higher level contexts out of these. MUPE cannot be run standalone in a terminal, as it is dependent on the MUPE server in the network, where all application logic is processed.

The two platforms highlight different aspects of context-awareness. Both personal (ii), and environment assisted context information (viii) were used. Many applications were created and tested with these platforms (ii, iii, iv, vii, viii), and MUPE is being actively used and developed further.

Chapter 3

Combining the Real and Virtual Worlds

3.1 Introduction

Context information allows the merging of real and virtual worlds. When something is known about the user's situation, a window to the virtual world can be opened that contains interfaces to virtual objects that are relevant to the current context. At lunch time restaurants can be proposed, in a noisy place the loudspeaker volume of a mobile phone can be turned up, and at a certain location, the nearby services can be shown, to name a few examples.

The previous chapter dealt with the technologies that are needed to retrieve the context information, and this chapter looks at how the user sees the combination of the real and virtual worlds based on the context information. This chapter deals with the last step in developing the applications, that is, the User Interface (UI) and Human-Computer Interaction (HCI).

The requirements for combining the real and virtual worlds are simple in theory as the user just needs an access to the virtual data on top of the real world. The virtual data is selected based on context criteria, and there can be many data sets based on different context criteria. The user has a window to the digital data via a device, through which the user can access and manipulate the digital data. The way the user is able to access the data is highly dependent on the used platform. With a simple device there is no way to create a complex UI nor is it possible to effectively manipulate the digital data.

There are several fields of research studying the combination of real and virtual worlds. Milgram and Kishino present a taxonomy on visual displays that look at how the real and virtual worlds are combined [69]. The complete field of combining the realities is referred to as Mixed Reality (MR). The Figure 3.1 shows the research area of Mixed Reality.

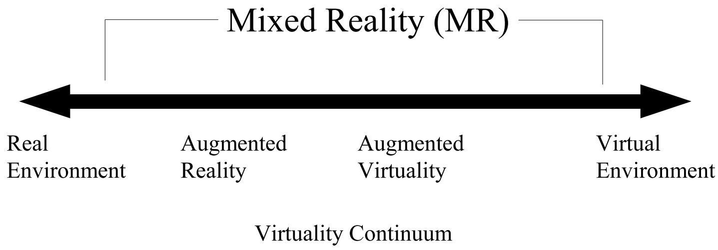

Figure 3.1: The areas of Mixed Reality [69]

At the ends of the mixed reality are the real and virtual worlds. Virtual Reality (VR) replaces the real world with a virtual world. In between, there are the research areas that mix the two ends. Augmented Reality (AR) supplements the real world with additional information, whereas Augmented Virtuality (AV) is the converse case, which augments a virtual world with real world information. At the left end of the continuum, the user is purely interacting with the real world, which is coupled with digital information. As we move right, the real world is first augmented and then replaced with the digital data. Most applications are at a specific point in the continuum but there are some that move in the continuum, such as the MagicBook [16].

Tangible User Interfaces (TUI) introduced by Ishii and Ullmer, is one area at the left end of the continuum [45]. TUIs have physical objects in the real world, which are coupled with digital information. Sensetable is one possible platform for building TUIs [72]. Sensetable is a tabletop workspace, which contains displays and physical objects for input, and the platform has been used succesfully in several research projects. The TUIs are presented and a Token+Constraint model for tangible interaction is proposed by Ullmer et. al. [98]. There are also commercial products available utilising TUI, such as Lego Mindstorms, which bring Legos into the digital age [123].

Augmented Reality referes to technology that overlays digital data directly onto the real world. This is usually done by opening windows to the digital world, and aligning this precisely with the real world. There are many options in implementing AR UIs, mainly with Head-Mounted Displays (HMD). Touring Machine, for example, implement an AR application that overlays digital information onto the real world utilising an HMD [32]. Other options include handheld devices, such as the ARToolKit on a PDA [100]. AR applications were already presented in the previous chapter.

Augmented Virtuality adds real world data into a digital environment. This area has been succesfully used in Computer Supported Cooperative Work (CSCW), where digital meeting participants can be augmented with live video or audio data. MASSIVE was an early example of such a system, and today most PCs include meeting tools for integrating video and audio [40].

Virtual Reality replaces the real world with digital information. This is the most used area of these, as most digital games implement some kind of a virtual world. The most common VR applications are highly visual experiences, where the real world is replaced with a computer generated 3D view. An early example of a system capable of visual VR is the Virtual Environment Display System [33] developed in Nasa. An earlier example of VR systems are the Multi-User Dungeons (MUD), and the first one was developed in 1979 [109]. The first MUDs used text-based interfaces and were not visual immersive 3D environments, although they were virtual environments. They still offered a rich immersive virtual world, that was mostly achieved with the users imagination.

A location-based application can be made with any MR technology, but they are quite different in nature. On the left end, the user can interact directly with real objects in the environment, whereas on the right end, the user is using a virtual world and not interacting directly with the real world.

Whenever we are dealing with location-based services, one can argue that we are never in a pure digital virtual reality, as the virtual reality already contains reflections of the real world. Some information in the virtual world is augmented with real world information, such as the location of real world objects. One example of location-based services are the digital maps that are a digital view to the environment. If looking at a pure map, it is in virtual reality, but when a user location is reflected on the map, it becomes augmented. Also, if we are at the left end of the MR continuum, we might not have any digital information visible, but we are still dealing with objects at a known location.

The MR continuum presented by Milgram focuses on virtual displays, but there is a lot more to merging the real and virtual worlds in location-based applications for a mobile user [69]. Location-based applications are used on the move, so the users of LBAs do not have technology that exists in desktop computing available. The devices set the requirements on how the real and virtual environments can be merged. A device with a few keys, small display and only limited processing power is not capable of creating AR applications overlaid on top of the real world.

3.2 Using location to combine the real and virtual worlds

Location-based Mixed Reality applications are a subset of context-aware applications that only need to look at one specific context, that is, location. As we are creating another layer on top of reality based on location, it is beneficial to use the term "world" to refer to this digital reality as well. The real and virtual worlds are somehow aligned, and windows to the digital information are opened. UIs of LBAs can be at any point in the MR continuum, depending on the used device and application. This dissertation deals with a limited spectrum of the continuum, as we only cover interfaces that are not at the ends of the spectrum.

The user moves in the real and virtual world at the same time, and this opens up many possibilities for the application developer. The virtual world can be a digital copy of the real world, such as in for example navigation applications, since the focus is on helping the user navigate in the real world. Other applications, such as games, can create alternate realities on top of the real world. These applications do not try to model the real world but rather use the real world as the container for their reality.

Location-based applications create an extra layer of information on top of reality. There can be many such digital realities that each represent a different set of information or a different application. A location-based application combines these realities in some way and creates an interface for the user to access these layers of information. Figure 3.2 shows the idea of the different layers. The figure contains a single map, which represents an area of the real world. On top of this, there are many layers of digital realities, which are sets of location-based data, or sets of location-based applications.

There are now many options for creating views into this world. On the right end of the MR continuum, a virtual world is created which is combined with real world information, such as the work by Rakkolainen et al. They use GPS together with Virtual Reality Modeling Language (VRML) [78]. As we move left towards augmented virtuality, the virtual world is filled with added real world information, such as location annotated video recordings or photographs (see, for example, [97]). Further left is Augmented Reality, where users act in the real world, but have windows to the virtual world. At the left end there are tangible user interfaces. These interfaces allow natural

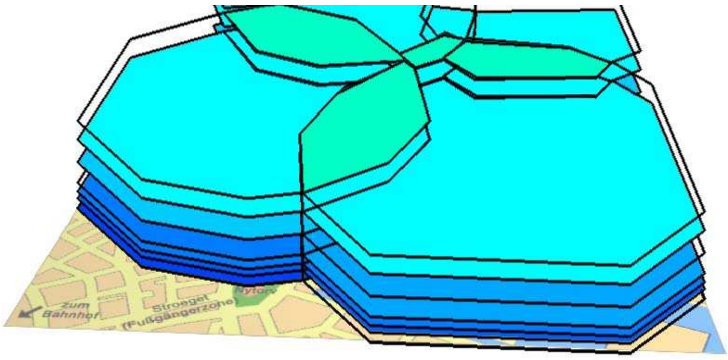

Figure 3.2: Layers of digital realities on top of the reality. Image by Jouka Mattila.

interaction by touching the actual objects, such as with applications done with the Sensetable [72].

The different approaches described here each offer a different way to interact with the virtual world. Each approach can use virtual objects that are manipulated with some input devices. Input devices can range from physical objects to traditional desktop UI devices like mouse and keyboard, to wearable devices such as N-Fingers [59], Tinmith head and hand tracking [75], or Twiddler2 [119].

3.3 Building realitites on top of the real world

A location-based application essentially creates a virtual world on top of the real world. There can and should be many virtual worlds, as a single digital world cannot fulfil the needs of all applications and users. Tourists, car drivers, bird watchers, and gamers are all interested in different things, but they can all co-exist on top of the same real world.

The LBAs can be roughly divided into two categories based on what kind of an application they try to create:

1. Applications that model the reality accurately as it is now. These applications use an accurate representation of the real world, and augment the reality with digital information. Maps, and other services that provide additional information of the real world are examples of such applications, like Cyberguide [3] and WalkMap (iii).

2. Applications that create an alternative realitity on top of the real world. Such applications do not recreate the real world in the digital domain,

but rather they use the real world as the container onto which they embed the alternative reality, or they use the real world information as an input to the virtual world. Examples of such applications include location-aware games such as BotFighters [110], or games presented in (viii).

It is not clear whether the recreation of an earlier era is real or alternative reality. On the one hand, the environment is real with a temporal shift. On the other hand, these realities do not reflect what is here right now and they are alternative reality. Such applications are documentaries or fiction of earlier times and they can be seen as alternative realities.

Figure 3.3: The real world overlaps with many virtual realities. Image by Jouka Mattila.

Figure 3.3 shows the idea of different realities on top of the real world from the user's perspective. The same real world acts as a container that is embedded with landmarks. The virtual worlds of the applications can reflect the real world, or they can create alternative realities. People in the real world are interested in different things, so they are most likely part of different location-based realities, although these experiences can also be shared. The real reality LBAs provide additional information relating to the real world. Whether the application is one of entertainment, tour guide, or maintenance assistance, they all have landmarks that are meant to be part of the real world. The alternative reality approach has much looser coupling between the real and virtual worlds. These applications use the real world, but try to create a feeling of being elsewhere. BotFighters places the players in the future where they control robots and engage in combat, while still using the real world as the playing arena [110]. ARQuake is another example, which creates the popular Quake game in the AR domain [94]. The monsters of the game are projected onto the real world, and the players are required to shoot the monsters in the real world. Pirates!, creates an alternative reality, where the real world is used as the playing arena, but the game world has no direct counterparts in the real world [17].

In the sense of technology, there is not too much difference between these two. The alternative realities have one advantage since they are not meant to perfectly align real and virtual information, and so they might manage with less accurate sensors. This, however, is not always true and is a matter of application or game design. If a game is designed so that, for example, the accurate distance between players is needed, the sensor requirements are higher.

3.4 Location-based mixed-reality application requirements

There are many possible platforms for creating location-based applications. The minimum technology requirement for all applications is that they get the location information with some method. The different applications can use different UIs which might have additional sensor requirements, and these are the differentiating factors for the applications. The additional sensors are needed, if the real and virtual worlds need to be aligned with great accuracy, or if advanced interaction is required.

The MR continuum shows that on the left side, where the AR resides, the applications need to align the real and virtual worlds precisely, as the main focus is on the real world. At the left end, the interface between real and virtual are in actual tangible objects, which does not involve complex alignment of real and virtual worlds. The AR on the other hand, aims at overlaying the virtual world on top of the real world, and this is a very complex task. There are two main approaches for AR [6], the first is video-AR [112] and the second is see-through-AR (ii), and there are also hybrids that use both methods [124]. In the video-AR, the real and virtual worlds are combined in or with the help of a video stream. With the see-through-AR, the user viewpoint is measured with sensors and only the virtual objects are shown on a see through display.

Moving towards the right in the continuum, there is less and less requirements for the alignment of the two worlds, as they become more separate. In a complete virtual world, the real objects are shown in their respective positions in the virtual world based on the location. This has much less requirements for the applications, as the simplest virtual world would present the objects that are nearby, which requires only a rough filtering of virtual objects based on the distance.

In this work, we divide the location-based applications into two categories based on their requirements, which were introduced in (i). One can argue that there could be many other divisions, or a completely continuous one, but for the purposes of this dissertation two is enough. The high-end and standard applications have very different characteristics and requirements. Still, similar solutions can be applied to the same problems provided the applications are properly designed. The two categories are:

1. *High-end location-based application* accurately merges the real and virtual worlds in real-time. These applications align the real and virtual worlds accurately, and present information from both at the same time to the user. This category includes the Augmented Reality applications that overlay digital information onto the real world with great accuracy.

2. *Standard location-based application* uses information from both the real and virtual worlds, but does not try to accurately align these two domains in the user's view. The real and virtual worlds can be aligned, but the accuracy is not crucial to the application functionality, nor is the information necessarily presented to the user at the same time. Maps are an example of these applications, since they represent the area the user is in without merging the realities.

The high-end approach to location-based applications requires a lot of processing power. The high-end applications need good input/output capabilities enabling complex interaction with the virtual objects, and the merging of the real and virtual views. The division of high-end and standard applications has a direct effect on the used hardware. The high-end applications require high-end platforms that are capable of matching the real and virtual worlds in real time. The standard location-based applications have much less requirements and they can be built with much simpler devices.

The discussed division separates the most complex applications from the rest. The AR applications offer intuitive interaction possibilities with the real world, but they are the most complex to implement. There can be applications that can fall into both categories. For example, an application can offer both an AR view, and a map based view to digital data. Such an application is still a high-end application if all of its features are used. If only the map is used, it can be used in a device capable of standard applications as well. There could also be more categories than the two presented here, or even a continuous categorisation system. For the purpose of this dissertation, it is enough to separate the high-end from the rest, since the high-end is so different in nature to others.

The division used here is based on our work on categorising location-based applications (i). In the paper, we present how the complexity of the location-based applications is dependent on how the environment is modelled, and how the user views the digital information. An LBA can model the environment from full 3 dimensions to no dimensions at all. The information can be viewed from the first or third person perspectives. The first perspective means that the location-based data is placed on top of the real world, as viewed from the user's perspective, as in AR. The third person perspective means the user sees both the location-based data and him/herself on the screen. The 1st perspective view to a three dimensional environment is most complex, and it requires the most processing power. This category includes the high-end location-based applications, that is, mobile AR applications.

High-end LBA device

A high-end LBA device is capable of creating high-end applications, which merge the real and virtual worlds accurately, that is, they know the location of the user and the view orientation of the user accurately so they can align the real and virtual worlds and present them together in real-time. To create such applications, a powerful device is needed. There are many possible platforms for creating such applications, such as handheld devices [37, 113], and wearable computers [32, 79, 96, 105] (ii-iv). The wearable computer used in our research can be seen in Figure 3.4. A wearable computer is not limited by size as the processing power can be distributed over the wearer's body. Cars already have commercial location-based applications available, as cars have enough space and power to sustain such high end applications. In-vehicle navigation systems have been around for a long time, and Head-Up Displays (HUDs) provide navigational aids while keeping the operator of the vehicle focused on steering the vehicle. NASA Human Centered Systems Lab offers a lot of resources and material concerning the HUD research [121].

This dissertation focuses on a mobile user, that is, a user walking alone and having a computing platform readily available, and this excludes cars. Cars can be seen as ubiquitous computing, as they are the environment with which the user is tightly coupled. The device needs to be carried by the user in order to use location-based applications wherever. Potential platforms for such applications are PDAs, mobile phones and wearable computers. Wearable computers have a lot of processing power and I/O capabilities, which make them good platforms for AR applications. Recent work by Wagner et al. proves that modern PDAs can be used for high-end AR applications [99]. Mobile phones have the least processing power, but high-end AR applications should be soon widespread, as many phones have cameras that can be used in such applications. Rohs presents how visual codes allow the mobile phone to interact with the real world [82]. The paper also proposes an interesting way to use the phone to interact with large displays.

The input capabilities of high-end LBA devices can be very powerful. Constructing 3D geometry outdoors requires a complex device, and the task is very much assisted with the help of a high-end LBA device such as a wearable computer [76]. Capturing the scenes could be done with normal cameras, but a high-end LBA device with powerful input devices allows the entire task to be done outdoors. The results can be immediately verified agains the real object that is modelled

High-end LBAs can be built with many technologies that are capable of determining the user view orientation and location very accurately. They need to track the user view orientation with either sensing the head orientation, or by using video cameras to know where the user is watching. The accuracy of the location depends on how far the augmented objects are; the closer they are the more accurate the user location needs to be known.

Standard LBA device

A standard LBA device is capable of creating applications that know the location of the user, but it is not capable of accurately merging the real and virtual worlds. For example, a digital map shown on a device only needs to know the location of the user, and even that can be inaccurate. The map of the surroundings can still be shown, and the user needs to do all the work in matching the digital information with the real world.

Such applications have very little requirements. They only require some means for locating the user, and even this can be inaccurate, or based on user input. The applications can use more precise location information, but as the realities are not merged accurately, it is not necessary to know the user's viewpoint accurately. The input capabilities of the device do not need to be complex either. Buttons, pen-based input, and voice UI are all sufficient depending on the application.

As there are not too strict requirements for the user, applications can be built with many devices. For example, a mobile phone with a GPS, or other positioning method, can be used for such applications. The application does not necessarily need a graphical representation of the surroundings, in some cases a simple list of objects is enough (i).

Figure 3.4: An example of a wearable computer. Photo by Tero Hakala.

3.5 Summary

This chapter presented the UIs possible for a location-based applications. The virtual world can use any measured data from the real world and update the object positions in the virtual world accordingly. A straightforward way to combine the real and virtual worlds is to use the location information, as this can be used to create a one to one correspondence between the real and virtual worlds. There are other options for creating the virtual world, and an alternative reality approach does not try to accurately recreate the real world. It is up to the application logic to determine, how this matching is done. Once the correspondence between the real and virtual worlds is known, an appropriate UI can be build.

Taxonomy for visualizing location-based information (i) presents a theoretical model for analysing the complexity of the UI used in a location-based application, and it helps in determining how complex UIs a certain platform enables. Two main categories of location-based UIs can be used. A high-end UI uses a first person view to a 3D environment creating an AR UI. Other kinds of UIs are simpler to implement and have less hardware requirements, which make them available for a larger number of devices.

In addition to the theoretical aspect, we have developed multiple different UIs for location-based applications. Context compass (ii), WalkMap (iii, iv), and MUPE games (viii) each show different kinds of mobile location-based UIs. The WalkMap and ContextCompass UIs are AR interfaces, whereas the MUPE game use a virtual world, that is augmented with real world related location information. All of these UIs can be mapped to the categories in (i), and the UIs use different perspectives (1st, 3rd) and different environment models (3D, 2D).

The interaction model of AR is very intuitive; placing objects at their correct locations in the real world is intuitive, but the AR applications have high requirements from the hardware. To reach a large number of potential users, a device that is part of user's everyday life is better, such as a mobile phone. In our research, we have used both aspects, each focusing on their strong features. Wearable computers focused on high-end location-based applications and mobile phones on standard location-based applications.

Chapter 4

Context-Awareness plus Mixed Reality implies Location-Based Applications

4.1 Introduction

Location-based applications combine the real and virtual worlds in a specific way. Real and virtual objects are combined by using the real and virtual locations of the objects, and a UI somewhere in the MR virtuality continuum [69] is constructed. Based on the platform capabilities and the application requirements, many kinds of applications can be built ranging from full AR systems in wearable computers to simple SMS based services on entry-level mobile phones. The applications may look very different, but the underlying principles rely on positioning. The UIs can be application specific, or different UIs can be constructed to the same location-based data, (for example (iii)). Even different platforms can access the same location-based application data, and they construct their UIs based on their capabilities. ARToolKit, for example, is implemented both in wearable computers using HMDs [112] and PDAs [100].

Location-based applications differ from the applications in the desktop world. The biggest difference is the situation of use. PC applications are used in a static use situation where the person sits in a static location most of the time. Laptops are already a different case, as they can be used on the go, and the use situation can be very different between the usage sessions, or if using in a moving location such as the train, the environment outside the train changes all the time. If a user can anytime and anywhere easily access location-based services, then it is possible that they become an integral part of everyday life.

The process presented for a context-aware application seen in Figure 2.1 can be directly applied to location-based application. First, the location is determined, next the information sources are combined and context data is formed, and after that the location information is provisioned to the actual applications.

This chapter presents how location- and context-awareness presented in Chapter 2 and MR UIs presented in Chapter 3 together form location-based applications. A context-aware platform together with a MR UI forms a location-based application.

4.2 The structure of a location-based application

This chapter looks how to build a location-based application. The building blocks of a location-based application is roughly composed of four main parts:

1. **Position** is determined with a positioning technology.
2. **Virtual objects a.k.a. landmarks** are retrieved from a landmark database.
3. **Application logic** is implemented into the device or network.
4. **User interface** is selected for the specific applications.

In the following, we address these issues in more detail.

4.2.1 Position data

To build LBAs, the location needs to be determined in some manner. There are two kinds of location information, according to Hightower [42]: Physical position and symbolic location. Physical position provides a coordinate point in a reference system, such as (Longitude, Latitude, Altitude) provided by the GPS system. Symbolic location has presents an abstract idea of where an object is, for example "in the railway station", "above me", or "in the airplane". One kind of information can be changed to the other, provided there is information available. For example, a database can contain objects at certain geographic positions, in which case we can transform a physical position to a symbolic location such as the "Railway Station".

The location system can be absolute or relative. An absolute system uses the same reference system for all locatable objects. GPS is an example of this, since receivers report their position with the (Longitude, Latitude, Altitude) notation. A relative system places the locatable objects in relation to themselves. For example, proximity based scanning can tell how far and in what direction another locatable object is from the initiating device.

There are two main methods for determining the position of a device, as pointed out in Chapter 1 (page 4): In **Infrastrucute assisted** positioning, an external infrastructure determines or assists the determination of the position of the locate object. These systems can be subscription based from a service provider, such as cell-based positioning provided by a mobile phone operator. In **Device determined** positioning, the located object detects its own position. For example, an object uses an attached GPS module for getting location information.

The first one is closely related to Ubiquitous Computing, in which the user device works together with the environment to create a seamless user experience. This method requires a tight coupling of the user device with the environment, and the device is not as well operable when far from infrastructure. A good example of this is WLAN network that can be used close to the base stations, sometimes referred to as hot spots. Also, 2G cellular technologies (for example GSM) are available in most places, whereas the coverage of the next generation 3G networks is much smaller.

The second approach is the one used in wearable computing, that is, the user carries the sensors allowing the user to take the applications where ever, whenever. This approach is highly mobile, but also requires the user to have all the sensors and processing power somewhere in one's body.

There are many devices capable of determining their location, and there are many widely used methods available that use both the infrastructure assisted, and the standalone version of positioning. GPS is the most widely used standalone technology, whereas cell-based positioning, such as E-OTD [100] and WLAN positioning, for example Ekahau [107], are widely used infrastructure assisted technologies. Nokia Wireless GPS Module LD-1W [122] is an add-on device that can be used with many different Bluetooth enabled Nokia phones, and in the future the GPS will be built into devices, as regulations such as "Enhanced 911 - Wireless Services" demand this [114].

Ekahau positioning uses WLAN to determine the position of the devices [107]. The system requires a client in the device to be tracked, and it has the accuracy of up to one meter. E-OTD and other cell-based positioning systems are available for the cell-phones [2]. These methods have the advantage over GPS in the coverage, the cell-based positioning works wherever the device has a connection to a base station, and with the mobile phones this covers most areas, also indoors. The accuracy, however, is not as accurate as with GPS. Cell ID based positioning accuracy is from 100 meters upwards to kilometers, and it is highly dependent on the density of the cell network.

As already discussed in section 2.1, the position information can come from many sources. A mobile phone, for example, can receive position information from a GPS module, and from cell-based positioning system. These can be combined in many ways, but with positioning, there is usually a rough estimate of the accuracy available. This can be used to determine which information to use, or an application can be designed to only utilise a specific positioning technology. With GPS, for instance, this means that the application does not always receive the signal.

In our research, we have used both device determined and infrastructure assisted positioning. The wearable computer applications (ii,iii,iv) used device detemined positioning, as a GPS was attached to the wearable computer. Infrastructure assisted location was used in (viii) where two different location producers monitored the players and generated position information. Our wearable and mobile phone applications also shared the location information between multiple users (v, viii).

4.2.2 Landmarks a.k.a. virtual objects

We have used the term virtual object to refer to context related data, but in location-based applications, the term landmark is often used [1]. The landmark represents an object in a known location. Such objects can be for example road signs, advertisements, stores, mountains or any other real or virtual data. Lynch [63] defined landmarks, as being physical objects of varying scale. The user uses landmarks to orient him or herself and the landmarks are varying in size and visibility, such as a tree in the park, or a skyscraper. In this dissertation I use landmarks and virtual objects and landmarks to refer to the same thing, but their origins is different: the term landmark originates from the real world, whereas virtual object originates from the digital world.

Maps can also be though of as digital landmarks, spanning a known area. Other frequently used terms in addition to landmarks are Place Of Interest (or Point Of Interest) (POI), and waypoints. POIs are similar to landmarks, and waypoints are landmarks with a special meaning. They mark the destination point, and possible intermediate points, of a route. Navigation from point A to point B can require many intermediate steps, and all these end points and in-between points can be marked with waypoints.

Landmarks can be stored either in the device itself, in a network storage, or both. The network storage can be a single massive collection, or it can be distributed into many specialised information storages. The landmarks can be divided into categories with metadata information, which makes it easier to select the interesting data needed. Personal landmarks, that is, landmarks that the user considers the most interesting ones, can be stored in the local device. The personal landmarks can also be shared between people or groups.

Many location-based applications are tailored to a specific service. For car navigation, it is not essential to know or present too much details on the surroundings in order not to distract the driver, whereas a walking user in a city is not interested in the petrol stations or petrol prices. Landmarks provide the essence of a location-based application, as they contain all of the differentiating information used in the application. By selecting a proper landmark set, the same application can be used for different purposes. Metadata associated with location-based data is essential in filtering.

One application does not fit all users. As seen from the above example of a car driver and a walking user, people have very different priorities depending on what they are currently doing. There should be easy mechanisms to customise applications for the users, and allow the users themselves to customise the applications. An application or service provider can make an initial offering, but many people like to customise their applications further.

There are two kinds of landmarks: static and dynamic. Static landmarks do not change their location or content over time, or change it very slowly. Examples of static landmarks would be buildings, cities, and rivers. A building is static, but what the building has inside is not: if we consider a museum, the museum itself is static but not its content. The museum has many exhibits in a year, and all of these are dynamic content. Which information is more interesting depends on the application at hand. As a tourist in a new location, a user can be most interested in the static, and well known landmarks of a location, such as the Eiffel tower in Paris, or the Big Ben in London. These locations are famous in themselves, and tourists are usually interested in seeing all of the famous landmarks. In this case, the user is at a new location, which makes the static landmarks also new. The dynamic content is the most interesting to a user in a static location. If we consider a user who is in his home city, the static landmarks become familiar quite rapidly. Such a user does not require help to locate landmarks in the city, but he or she might like to have the dynamic information readily available. For example, a museum exhibit changes many times a year, and to have such information at hand might be the key factor when the user makes a decision to visit an exhibit.

In our research, we have used both personal and shared landmarks. Our wearable computer (ii, iii) experiments were mostly about single user applications, where the landmarks were stored locally, and did not require a network connection to access. The landmarks could be shared, as was done in (v). MUPE platform is designed for multiple users, and the landmarks in it can always be shared, as was done in (viii).

4.2.3 Application logic

The application logic of a location-based application can be implemented either standalone, or distributed in the network. Any software architecture can be used, such as a multi-tier architecture. Location-based applications are like any other applications, except they are built on positioning and landmarks.

The positioning might not be always available, as the location sensor could be inoperative due to special conditions or the connection to the infrastructure might be offline (e.g. out of cell coverage with a mobile phone). These situations can be dealt with many ways, and the key factor is the duration the positioning is offline. If the positioning is offline only for a few seconds, no real damage is done. For a walking user, the environment does not change too much in a short period of time, and even in a car, it can be estimated at what speed and what direction the car is going. If the offline situation is on for a long time, the application needs to expose the situation to the user, as a reliable estimate cannot be made. This is, however, again a matter of application design. Applications with slow update rates do not need the position information to be up-to-date all the time.

Landmarks are the other special feature of location-based applications. If landmarks are shared over a network, the disconnection makes the update of landmarks inoperable for that time. Positioning can be operable, and the application can seem to operate without any glitches. However, if landmarks are constantly updated, for example the landmarks are retrieved based on the current position, after a certain period of time the application does not have any landmarks in the current surroundings. This again requires the application to expose the situation to the end-user. If landmarks are all stored in the device, this situation does not exist. A standalone LBA is more error resistant, but the data is not necessarily up-to-date. One solution for combining the two choices is having a standalone database in the device, which is updated from a central networks storage when needed.

Application logic can also be distributed. A client-server or peer-to-peer architecture have many desirable features, but such systems are also more dependent on the network connection. A location-based application is meant to be used by a mobile user, who by definition is in different locations at different times. It is beneficial that the user would have the possibility to use the resources available in the current environment, and a distributed architecture is very useful in many situations.

Application logic is responsible for mapping the objects in the real world to the virtual world, and determining the behaviour of the application. As discussed, the application can recreate the real world as it is, or create an alternative reality on top of the real world. Also, many different kinds of applications can be built with the same landmarks, such as tourist guides, or personal memory assistants.

Our wearable platform had all parts in the single device, making it error resilient. The single-user applications (ii,iii, iv) were operable even when the network disconnected, but the extension that allow the sharing of parts of the applications (v) was unoperable if the network went down. MUPE platform and applications (viii) had the application logic and landmarks in the network server, while positioning and UIs were in different places. Such a system could operate even if the positioning component was inoperable, but without up-to-date information.

4.2.4 User interface

The final part of a location-based application is the UI. This part is the most relevant from an end-user perspective, as it determines how the real and virtual worlds are combined, which is essentially the idea of an LBA. Many different UIs have been proposed some of which have very high requirements of the device, while others can be used in very simple devices.

The taxonomy (i) analyses how the user interface can be built. It shows that the UI can be a complicated AR UI, or a very simple UI that can be accessed with an Short Message Service (SMS). The simplest application would only present a list of objects in the surroundings, whereas the most complex appliction would overlay the information right on top of the object in the real world. Still, the most common application is a map application. Map applications are not too complex to implement, and they work well with less accurate sensors.

The standard applications that do not need complicated hardware can be run on ordinare everyday devices, like mobile phones and PDAs. Benefon Esc! is a mobile phone that has a GPS and a complete map service built in [113]. Many more devices will have GPS in them in the future, due to many requirements such as the E911 [114], which requires that mobile phones making an emergency call need to be positioned to provide better and quicker help. Palm PDAs can have a GPS module attached to enable the map services [118] and similar solutions are available for mobile phones, see for example [122].

These applications, as they are used in small devices that fit into the pocket, have moderately good I/O capabilities. A typical device has a small display and sound for output. The input almost always has some keys, and sometimes a touchscreen and voice input are also available. The voice input is subject to varying noise conditions experienced by a mobile user.

Point and click UIs are very intuitive to use. They allow the user to directly point at a UI object, and this is the most dominant UI at the moment. Point and clicking is a natural way to develop an LBA as well, but there are more ways to use this technique. Point and click can be extended to the real world (ii) as well, where the user points in the real world. The simplest LBAs do not implement real-world point and click, but more complex applications use this advanced technique. GeoVector is a company that has many real world point and click applications ready to be commercialised [116].

Moving towards higher-end applications, i.e. AR applications, the hardware requirements increase a lot. High-end AR needs very accurate positioning and a very accurate view orientation knowledge, or the use of a video camera and a lot of image processing. AR applications use an accurate 3D model of the real world, and processing the model consumes a lot of processing power. Many AR applications are used with an HMD, which makes additional requirements for the platform used.

Many AR applications have good I/O capabilities enabling complex interaction with the virtual objects. Good I/O enables the use of complex UIs (ii), but the hardware requirements usually result in devices becoming large, like a wearable computer [32, 61, 124]. There are other options for creating high-end location-based applications with for example PDA. The PDAs do not usually have as good I/O capabilities, and they are often required to work tightly with computational resources in the environment.

Mobile Augmented Reality (MAR) is Augmented Reality used on the move, and not in a restricted environment [79]. The MAR systems are carried all the time and used whenever needed. Such systems can be used to help a fieldworker [15], to help in navigation [3, 32, 79, 96, 105], and generally in collaborative work [14], games [94, 95] to name a few areas.

Augmented Reality's most serious problems lie with registration, that is, the alignment of the real and virtual domains [6]. If the worlds are not properly aligned, the AR loses the ability to point and click at the real world objects, and to provide information at the correct real world positions.

Sight is not the only sense that can be used. An audio user interface was presented in, which has no display device and relies on 3D audio [84]. Another non-visual approach was taken by Ertan et al. who used a haptic user interface [29]. These UIs might in many cases be preferable over visual output when dealign with mobile users. A walking, and especially running user needs to focus on the real world with the sight, and further, it is not easy to focus on a small digital output unit while on the move, especially a small screen.

Visual output is the only possibility for accurate millimeter scale alignement of the real and virtual world, but as already discussed in this thesis, this is not a necessity in LBAs. Relevant information can be provided for the user even with other senses. Many guidance systems provide aural guidance, and location-based reminders on mobile phones can utilise tactile feedback, as any other notification in the phones.

The main difference of a location-based UI compared to a traditional computer UI is that the user is interacting in the real world, rather than in a traditional computer desktop. This is a challenge both for standard and high-end applications. Simple location-based applications might require the user to constantly switch their attention from the UI to the real world, thus making the application cumbersome to use. AR applications have different problems, as the UI is constantly overlaid on top of the real world. There might be dangerous situations if virtual objects block out dangerous situations from the real world.

We have applied many different UIs in our research. Our taxonomy (i) offers theoretical means for analysing the complexity of location-based applications. We have created many different kinds of UIs: 3rd person view to a 3D scene (iv), 3rd person view to a 2D scene (iii, viii), 1st person view to a 2D scene (ii), and some other viariations. Each of these UIs offers a different view to the location-based application, which are suitable for different situations.

4.3 Location-based application categories

This dissertation has so far presented many different location-based applications. Some are available as products, while others are still under development by the research institutes. Friend finders [28], maps [113], tour guides [3], SMS services for location-based anonymous instant messaging [19], location-based notes [71] and games [111] are all available for the public, whereas augmented reality [64] is still a research topic, although several technologies, such as mobile ARToolKit [100] bring the technology closer to an everyday device.

Referring to the categories presented by Dey, context-aware applications can be divided into three different areas that are [25]: *presenting*, *execution*, and *tagging*. These categories apply also to the location-based applications. In addition to the division presented by Dey, there are many other divisions available to the location-based applications. They can be divided based on the complexity of the UI (i), what senses they are using for interaction, whether they are using an accurate representation of the real world or creating an alternative reality, and the task they were designed for to name a few.

Korkea-aho presented a context-aware applications survey in which there are categories for context-aware appplications [49]. Each application area presented in this survey, can be applied to location-based applications as well as location is an important context information. The survey used four categories: office and meeting tools, tourist guides, fieldwork tools, and memory aids. Azuma et al. propose three categories for AR applications: mobile, collaborative and commercial [6].

The biggest difference in location-based applications is whether they recreate the real world, or if the location information is used for rendering an alternative reality on top of the real world. If the application recreates the real world, it means that all buildings, objects, and geometry in the real world is used as it is, and additional information is provided in the form of landmarks. Alternative reality does not use the real world geometry. For example, a historical application can recreate an ancient forest in place of the current city, and in this case the user cannot rely on the virtual scene to be the same as the real world. In both approaches, there are landmarks in addition to the geometry.

The alternative reality approach is typical for games, as games can be fun without accurate matching of the real and virtual worlds. This does not mean that games could not use an accurate model of the real world. For example, Pirates! created an alternative reality on top of the real world, where the players roamed the sea, which was the alternative reality. The players also encountered islands and other players in the game arena [17]. ARQuake, on the other hand, uses an accurate geometry of the real world (campus area), and mixes game elements directly with the real world, with AR technology [94]. The game and real world geometry were the same, but the objects in the game world were game objects.

When applications use an accurate model of the real world, and embed the landmark objects into it, there are many typical application areas, as seen in literature: navigation [3] (iii), landmark browsing and search [32] (ii), information creation [71, 76], memory prosthesis [81], presence [19], surveillance [66, 88] and other real-time sharing applications. Surveillance has well-known problems of privacy, and Little Brother [18] is a good article discussing if the wearable sensors could increase security felt by elder people.

Table 4.1 lists location-based applications presented in this dissertation and some key features of their behaviour. Many more applications were

Table 4.1: Location-based applications and their features.

Application name	Type	Platform	real	alt.
Cyberguide [3]	Tour guide	PDA	X	
WalkMap (iii,iv)	Navigation	Wearable	X	
Context compass (ii)	Landmark browsing	Wearable	X	
Constructing 3D geometry [76]	Information creation	Wearable	X	
The wearable remembrance agent [81]	Memory Prosthesis	Wearable	X	
FriendZone [19]	Presence	Mobile phone	X	
Sousveillance [66]	Wearable surveillance	Wearable Computer	X	
Botfighters [110]	Mobile Game	Mobile Phone	X	
ARQuake [94]	AR Game	Wearable	X	
Pirates! [17]	Game	PDA		X
Rapid Prototyping with MUPE (viii)	Games	Mobile Phone	X	X

presented in this dissertation, and the list includes example applications by us and others that characterise the differences of the applications. The list shows that the majority of the applications recreate the real world accurately, but there is a lot of variance in the applications. There are also many different platforms used in the research, and the platforms are the deciding factor when building the UI. The wearable platforms are capable of AR interfaces, whereas other platforms usually do less fusion of the real and virtual worlds in the UI. there are naturally exceptions, as handheld devices can create AR interfaces, and wearable computers can use non AR interfaces.

The final entry in the table (viii) includes applications in both real and alternative reality columns. The paper presented four games and an application, out of which some recreated the real world structure of the location, while some only used location as an input for a different kind of experience.

4.4 Summary

This chapter looked at how location-based applications are put together. A location-based application is composed of four parts: Positioning technology, landmarks, application logic, and user interaction. These four parts are needed for every application, although many parts can be shared between applications.

Location-based applications include many possible application areas. Location-based applications exist in areas of navigation, tour guides, landmark browsing, search, presence, surveillance, and games. Our research concentrates on three main application areas: navigation, landmark browsing and games. Two platforms are also used: a wearable computer and a mobile phone.

Building location-based applications to such different platforms differs mostly in user interaction part. A wearable computer has a lot of processing power, good and extendable input devices and a HMD for visual information always at the field of view. A mobile phone on the other hand, is a fixed device that is most of the time in a pocket. The underlying application structure can remain the same and the differentiating factors are found in the UI.

All our papers focus on one aspect of location-based applications. Paper (i) presents a theoretical analysis on the UIs. Papers (ii-iv) are in the AR domain and they present a navigation application, and (v) presents a method on how to conduct User Experiments on such applications. Paper (vi) discusses how to develop tools for end user content creation to location-based applications and the themes in this paper are the design principles behind the MUPE platform (vii). Finally, (viii) presents how a solid platform (MUPE) presented in (vii) is used to rapidly develop new location-based applications into mobile phones.

Chapter 5

Introduction to the Papers

5.1 Overview

The major contribution in this dissertation is presented in the eight research papers. These research papers study the theory of visualising location-based information, what kind of infrastructure supports rapid development of location-based applications, and how to support user-created content. In this chapter, we will present each paper and how it is related to the big picture in this thesis. First, we will look into the platforms used in this study. In our research we used two platforms: a high-end wearable computer platform, and a standard mobile phone platform.

The high-end approach for location-based applications was studied with a wearable computer prototype. The wearable computer was built with off-the-shelf components, but it is still highly custom design, since wearable computers are not yet widely available as commercial products. Some products also exist in the wearable category, such as those provided by Xybernaut Corporation [126].

Our wearable computer uses a head-worn display for output, and N-Fingers [59], which is an interaction technique based on buttons on fingers, and a trackball mouse for input, as seen in Figure 3.4. A compass is attached to the headgear for determining the view orientation, and GPS for the location. The central unit is a small PC that runs for a few hours on a single battery. All these parts were integrated into a belt design, as seen in Figure 3.4 that was easy to wear on an average person. The belt contained most parts of the system, and a shoulder strap was made to get better GPS signals by placing the receiver on the shoulder.

The software architecture that was used in all applications was Message EXchanger (MEX) [58]. It is a small footprint architecture delivering messages between different components inside a single device, or between devices. The modular architecture of the wearable computer benefitted from this approach, and multi-device communications was utilised in the usability testing.

Four of the papers concentrate on applications built with the wearable computer. The applications revolve around *WalkMap* (iii), which is a navigation application for a wearable computer. The application helps a walking user when navigating in the environment. *Context compass* (ii) presents a minimal screen space approach for browsing and accessing information, and *The evolution of perspective view in WalkMap* (iv) presents how the maps can be enhanced for a specific task. We also developed a *system for evaluating augmented reality user interfaces in wearable computers* (v) that utilises a control computer for controlling the wearable computer in a usability test.

The wearable computer is good for researching high-end applications, but simpler applications can be prototyped with smaller platforms, such as mobile phones. The other major part of our research concentrates on location-based applications on mobile phones. We developed the Multi-User Publishing Environment, which is an application platform for creating context-aware multi-user applications in mobile phones. Three of our papers concentrate on this aspect.

The *Augmented Reality for a Casual User: Designing Tools for Interaction with the Virtual World* (vi) discusses how the user-created content is essential, when designing applications for a mobile phone. The *Open-Source Game Development with the Multi-User Publishing Environment (MUPE) Application Platform* (vii) presents how multi-user context-aware applications and multi-player games are created into a mobile phone. The final paper *Rapid Prototyping of Location-Based Games with the Multi-User publishing Environment Application Platform* (viii), looks at how to quickly create location-based applications on a mobile phone.

Taxonomy for visualizing location-based information (i) analyses how the UI of a location-based application is made. It clearly shows that simple location-based applications are enough for many tasks, and a high-end approach is not always needed. The paper is the backbone for the other papers, as the taxonomy can be applied to all the applications presented in this thesis.

In each application, there is also a schematic of the software organisation in each respective case, with respect to the big picture of the thesis provided in Figure 1.1. These schematics highlight the different approaches to the software in each application. Technologies in each cell of Figure 1.1 were developed in this dissertation: the UI was both in single and multiple devices. WalkMap used both local and distributed landmarks, and standalone application logic. MUPE uses a networked application logic and landmark server, although at runtime some of these are stored locally on the device. Both device determined and infrastructure determined positioning methods were used. These papers are presented next, with comments on the roles of each individual author in the papers.

5.2 Taxonomy for visualizing location-based information

Based on the work done with location-based applications, we developed a taxonomy for visualizing location-based data (i), which is a theoretical model for analysing location-based UIs. This taxonomy looks at the complexity of the environment used in the visualization, as well as the viewpoint of the user. Based on these, we formulate the (m,v) tuple (model, view), which tells how the application visualizes the environment. The model refers to the environment complexity, and can range from zero to three dimensions. The viewpoint is either a first, or a third person view.

The (m,v) tuple can be used to determine the complexity of creating a location-based application. The higher the environment complexity, the more processing power is needed. Rendering 3D geometry is much more complex than using a 2D model of the environment, or a list of closeby objects. Similarly, the first person view requires more accurate knowledge of the user orientation towards the environment, as the data needs to be aligned with the real world very precisely. This has a direct effect on the sensor accuracy, and the more precise information needed, the more accurate sensors are needed.

By looking at the data in the paper, we can see that on the one hand, creating an AR application that is a first person view to a three dimensional UI environment, requires a lot from the hardware. On the other hand, even an application with no environment model and any view perspective to the data, can be useful in some situations. An example of such a case is a list of people in the same location as the user. This fact suggests that a high-end approach to the location-based applications might not always be necessary.

This paper can be used as a starting point in analysing the rest of our papers. All location-based application UIs can be projected onto this taxonomy, and some estimates of the application complexity can be made with this. Next, our papers on wearable computers are presented, followed by the work on mobile phones.

This paper was written by me and Juha Lehikoinen. Both authors contributed equally.

Table 5.1: The taxonomy of visualizing location-based data based on the complexity of the environment model and viewpoint.

MV	Minimum sensor requirements (in addition to location)	Application complexity in terms of environment model
(3,1)	High (3D sensors)	Very high
(3,3)	None (a 1D sensor if forward-up is used)	High
(2,1)	Low (a 1D sensor)	Medium
(2,3)	None (a 1D sensor if forward-up is needed)	Medium
(1,1)	Low (none for distance-based; a 1D sensor for angle-based)	Low
(1,3)	None (a 1D sensor if forward-up is needed)	Very low
(0,0)	None	Very low

5.3 Context compass

Context compass is a method for retrieving, visualizing and displaying virtual objects in the user's environment (ii). The system is composed of three main parts: First, a set of virtual objects are retrieved based on context information. The context information sets filtering criteria, such as how far can the virtual objects be from the user in order to download them. Second, the user interface of context compass displays a preview of the virtual objects in the surroundings. The virtual objects are presented with a small icon, and they are placed onto a compass view. Third, the user can access the virtual objects by looking towards them in the real world. Context compass runs standalone in a single wearable computer, but it could also access a remote landmark container via MEX. Figure 5.1 shows how the context compass (ii) relates to the big picture.

Context compass is shown on a head worn display, and the design uses minimal screen real estate, as can be seen in Figure 5.2. The problem that was addressed with the design was the fact that the view of a walking user should not be blocked by too much information. The field of view of a user should be kept clean of AR data in order to not distract the user.

Context compass uses a traditional compass as the starting point. The traditional compass contains only the compass points (North, South, East, West), and these helped the user in navigation. In context compass, the linear compass shows a 360-degree range, the compass points and the additional

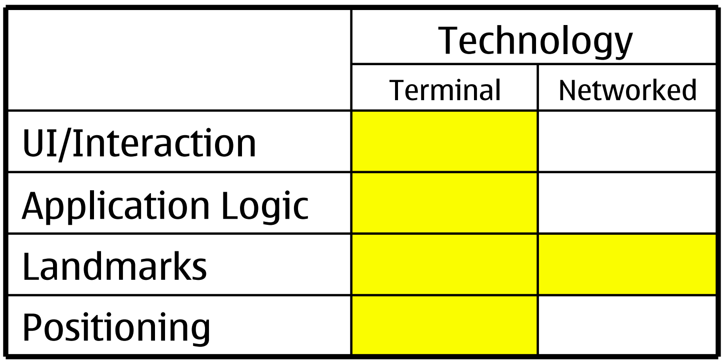

Figure 5.1: The software organisation of context compass.

points that represent virtual objects. These virtual objects are presented in their appropriate direction from the users current position, and the user can select these objects by looking at that direction. When the user has looked to the direction of the object, the object is highlighted and its name and distance to it are shown. The user can now access the object by pressing a button on the input device, and open data associated with the object.

Context compass is an essential part of WalkMap, namely when the user does not need to see a map of the environment. Context compass allows the user to walk undisturbed in an environment, and browse virtual objects to only use them as navigational aids.

In addition to this the paper, there is a patent pending on context compass [92], and a further study on the UI [60]. The hardware requirements of context compass are quite complicated, as it requires a compass in addition to the GPS. The UI mapped to the taxonomy is 1st person view, to a 2D environment model (also 1D is possible), i.e. mv(1,2).

This paper was written jointly by me and Juha Lehikoinen. Lehikoinen was responsible for most UI development, whereas context compass was my original idea and I implemented the system. The section with the design was co-authored. This paper also appears as a part of Lehikoinen's PhD [57].

5.4 WalkMap: Developing an augmented reality map application for wearable computers

The development work of the navigation application is put together in this paper (iii). The paper summarizes the different navigation and interaction methods, as well as looks at the iterative design process of the complete navigation system. WalkMap could run standalone as context compass, but

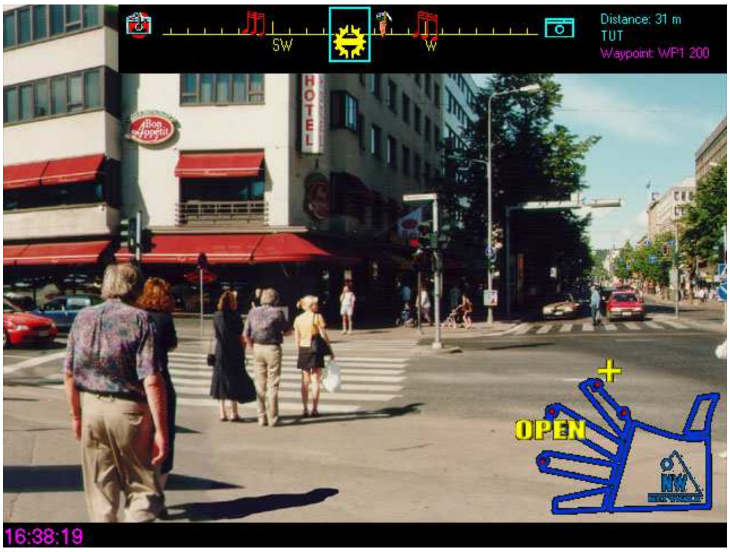

Figure 5.2: A generated screenshot of context compass.

it also accepted landmarks from a network container. Figure 5.3 shows the organisation of components, which is the same as in context compass.

	Technology	
	Terminal	Networked
UI/Interaction		
Application Logic		
Landmarks		
Positioning		

Figure 5.3: The software organisation of WalkMap.

WalkMap application is composed of several different visualization modes, which are suitable for different tasks. Context compass, already presented in paper (ii), provides a good see-through navigation mode but still allows a good view of the real world. The other visualization modes include maps that provide information about the surroundings, but still offering some see-through capabilities. The map presents the user with a digital view of the surroundings, which allows the user to know more about the environment than he/she sees in the real world. A screenshot of WalkMap maps can be seen in Figure 5.4. The paper also presents the software and hardware architecture of the platform, as well as Human-Map interaction, most notably the N-Fingers [59].

Figure 5.4: A screenshot of two different map views in WalkMap. Black is see-through area on the HMD.

The paper was written in close collaboration with Juha Lehikoinen, and each author contributed equally to the paper. I was responsible for the software development, whereas Lehikoinen was responsible for the user evaluations. This paper is also part of Lehikoinen's PhD dissertation [57].

5.5 The evolution of perspective view in WalkMap

The final improvement to WalkMap, a three dimensional map, was presented in this paper (iv). The 3D map allows even more possibilities for the navigation application, as well as posing new problems. The paper concentrates on the map visualization techniques, namely how should the map be visualized, to support navigational tasks, especially targeted search.

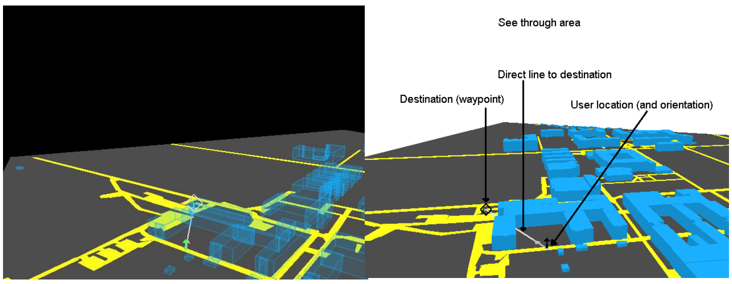

Figure 5.5: A screenshot of the evolution of the perspective map.

The paper presents the perspective map, its evaluation and the effect on the design. The perspective map [61], which is one visualization mode in WalkMap is not a 3D map, rather, it is a 2D plane viewed from the birds eye view. This is a bit restricting on how it can be modified, so in order to improve it, the 3D view was developed. 3D maps, however, present some problems of their own, most notably, the objects that rise above the ground level might block the view to objects behind them. This is also the case in real world, but in the virtual world we can design around these problems. The paper presents various solutions for these problems. In Figure 5.4 there are two views presented, and one view shows the buildings transparent, which allows the user to see the streets behind the building.

This paper was written jointly by me, Kimmo Roimela and Juha Lehikoinen, and all of us contributed equally to the paper. I was responsible for the overall software, and Roimela was the 3D expert and implementor of the proposed 3D solutions. The concepts for solving the blocking problem were developed jointly with the authors.

5.6 A system for evaluating augmented reality user interfaces in wearable computers

This paper deals with the difficulties in conducting usability research on wearable computers (v). A wearable computer is not a standard platform, and there are significant problems in the user research, namely how to see what the test subject sees and how to control the events so that the goals set for the testing are achieved. Furthermore, as all of this takes place while the user of the wearable computer is walking in the environment, how to support all of these in a mobile setting. The Evaluation software extends the WalkMap by distributing the application logic and UI among many computers, as seen in Figure 5.6.

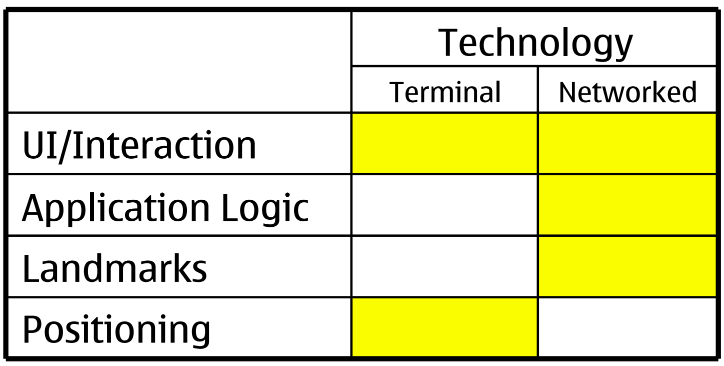

Figure 5.6: The software organisation of the evaluation software.

This work sets a goal to create a usability test platform that could be used with all the usability tests with the platform. The general idea is composed of two parts. First, the subject's UI was recreated into the supervisor's PC, and second, the supervisor had total control over the subjects location-based data set. The suervisor could constantly monitor what the subject was seeing, and thus could determine whether the subject's software had problems, or whether the subject made errors in judgement. The control of the location-ğbased data allowed the supervisor to set the pace for the test, and decide what is seen in the subject's UI. The setup for the tests is seen in Figure 5.7, where the test controller is following the test subject with the test control software.

Figure 5.7: The setup for the usability tests with a wearable computer. Photo by Tero Hakala.

The paper was written jointly by the three authors. The usability evaluation software was my initial design. Juha Lehikoinen helped set the requirements for the software, and Ilkka Salminen was responsible for the communication between the test subject and controller devices. The system was tested with WalkMap, and the tests were organised by me and Lehikoinen.

5.7 Augmented reality for a casual user: designing tools for interaction with the virtual world

In paper (vi), the focus is on end-user created content instead of the technology at hand. This paper discusses the ideas behind the Multi-User Publishing Environment, that is,

- The need for end-user tools
- The client to the world should be run on devices that are in users pockets already
- Give everyone the possibility to easily create their own context-aware experience on top of the real world

The target platform was designed to be generally context-aware, not just location-aware, as the triggering criteria discussed in the paper can be applied to any information source. The paper presents two sets of tools. First, mobile tools are meant for the end-users, who can create context-dependant content into the virtual world anywhere anytime. These small tools should be adaptable to the devices they have with them. Second, the high-end tools are meant for the virtual world developers, who can create entire virtual worlds with powerful desktop computers.

The paper discusses the low-end approach, that is, to use devices that are most likely to be found on a normal user's pocket. Most mobile phones today contain Java 2 Micro Edition (J2ME), which was the basis for our design. The high-end tools are implemented with Java 2 Standard Edition (J2SE), to allow best interoperability accross server platforms.

This paper was written jointly by me, Jouka Mattila, Eero Räsänen and Timo Koskinen. I wrote most of the paper while others assisted. The theme of the paper was developed jointly with the authors.

5.8 Open-source game development with the Multi-User Publishing Environment (MUPE) application platform

Multiplayer gaming and multi-user applications are a complex area in mobile devices, which is discussed in paper (vii). The paper presents the Multi-User Publishing Environment (MUPE) application platform, and analyses two different games made with the platform. The platform offers an easy way to extend virtual world that can be customised in many different ways. As seen in Figure 5.8, the application is not location-aware, since it concentrates on the multi-user aspects in mobile devices. However, MUPE is designed to be context-aware and the paper presents the context engine in MUPE, which supports any kind of context-awareness.

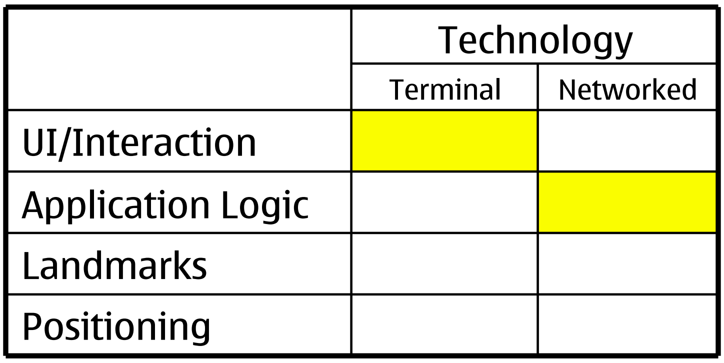

Figure 5.8: The software organisation of MUPE.

The focus of the paper is on the platform issues, which are the basis for all standard location-based application. Multiple users, general context-awareness and high latencies of mobile networks require a solid platform, which was the focus of this paper. The games presented in this paper were made over the HTTP protocol, which set strict limitations on what should be done and how. The HTTP does not allow real-time communications, as the player turns need to be synchronised in the game server. This affects how games should be designed, and how the turns should be taken. MUPE is the platform, where the location-based games presented in the final paper are built.

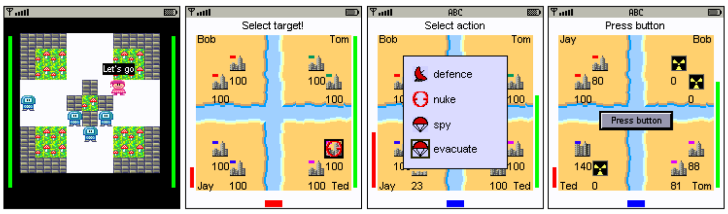

Figure 5.9: Screenshots of multiplayer games developed with MUPE. All screenshots are taken with the Java J2ME Wireless Toolkit 2.2 Emulator.

Two different games were presented, FirstStrike and MUPEDungeon, and screenshots are presented in Figure 5.9. MUPEDungeon is less dependant on the HTTP connection, as the user can make many moves without the need to wait others, and the game experience was mostly related to a single user's latency. FirstStrike, on the other hand, was much more difficult, as the game turns need to happen at the same time. Four players have a limited time to make a move, after which the server is updated and the new state is communicated to all of the users. This makes the game dependant on many latencies, and much more demanding when user experience is concerned. If some device has a high latency, every move will take a long time. Based on this paper, we have later implemented push communications using TCP connections that became available with J2ME MIDP 2.0.

This paper was jointly written by me, Eero Räsänen, Ari Koivisto and Jouka Mattila. I wrote most of the paper while others provided technical facts. The games were developed by Räsänen, Koivisto and me, and Mattila was responsible for graphics.

5.9 Rapid Prototyping of location-based games with the Multi-User Publishing Environment application platform

The context-aware application development is time and resource consuming. The Multi-User Application Platform, already presented in the previous paper, provides many ways in which the application development speed can be increased. Paper (viii) studies how location-based games can be developed with a platform offering ready made components, and this paper also shows how easy it is to develop location-based applications with MUPE. As seen in Figure 5.10, this paper extends MUPE into the location-aware domain, and it is our first paper to use infrastructure assisted location.

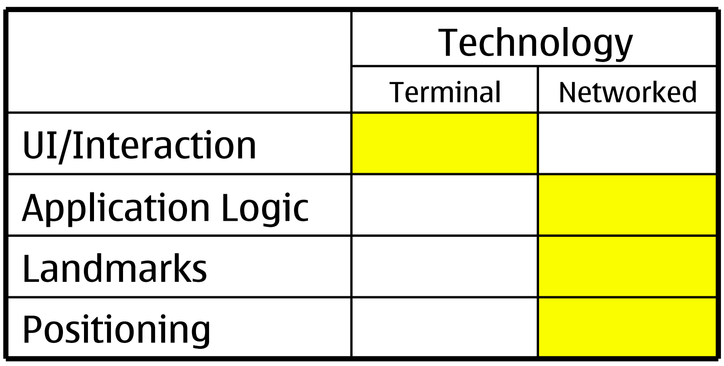

Figure 5.10: The software organisation of location-based games with MUPE.

A 24 hour coding session was organised in Lappeenranta University of Technology, where five teams were developing multiplayer games for mobile phones, during a fixed timeframe. Nobody was familiar with the platform beforehand, but everyone had a lot of experience in programming, also with Java, which is used in MUPE. Each team produced one location-based game design and implementation. The 24 hours were divided between an introduction to the platform, design, coding, and game evaluation by the organisers.

The session proved that with a solid application platform the application and game development speed can be increased considerably. With a rapid development platform it is easy to create prototypes of application ideas quickly, and make an initial evaluation of the applications. Figure 5.11 shows three screenshots of the games developed in the session.

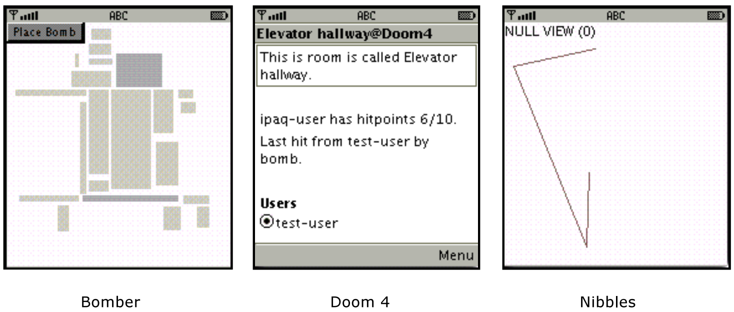

Figure 5.11: The software organisation of location-based games with MUPE.

The codecamp used all the themes presented in paper (vi): each developer group created their own AR experience, end user's created the world content with transparent tools, and the target client device was a mobile phone, which all the participants owned.

This paper was written together by me, Kimmo Koskinen and Kari Heikkinen from Lappeenranta University of Technolgy. Koskinen was responsible for the location extension to MUPE, while I was responsible for the MUPE server development. The test situation was organised jointly by all three authors.

5.10 Design recommendations

There are many lessons that can be learned from all of the applications presented in this chapter. The wearable applications are very easy to access and use, whereas the mobile phones are very easy to carry in ones pocket, and almost as easy to use.

All location-based applications have some interaction with the real world. Different situations require different techniques, and different users value different things. Often, the users are familiar with their surroundings and an always on UI is not needed, but when information is needed, the more complex the UI is, the less the user needs to do matching between the location of a landmark in the device UI and in the real world.

The accuracy of location can also be hidden behind the application design and UI. In navigation application, it is important to have accurate location information, and with our navigation applications we aimed at maximum accuracy. Since the accuracy was in the scale of ten meters, we chose the map scale to be such, that the YAH symbol covered this area. When developing entertainment applications, the design can take much more liberties. The games developed in our rapid prototyping session often used rooms as the location. Such location is not accurate nor remains the same as the accuracy varies according to the room size. Still, this accuracy is enough, since the location is perceived differently indoors.

The four layers that describe the location-based applications can assist the application developer. The first layer, the positioning, determines whether the device needs positioning technology. The second layer, landmarks, is memory intensive. The more data is stored in the device, the more storage is needed. Also, in case the applications are multiuser, this has to be at least partially networked. The application logic and UI can gain information from our taxonomy (i). It can be used to estimate the complexity of the application development from two perspectives. First, if the platform is known the possible application UIs can be chosen. Second, If a UI is chosen, the taxonomy helps in determining what kind of sensors are needed. Also, the designers should keep in mind that alternative reality applications have less sensor requirements than real reality applications.

The most important lesson learned from the application development perspective is the need for a solid platform for developing the applications. Wearable computer is not a standard platform, and a lot of work went into the construction of the hardware and software before the actual application development, and user testing could start. The same situation duplicated itself with mobile phones. Mobile phones have standard software development environments, but still the initial software development takes a long time. Application research should not be solely software development, and the focus should be on the applications. With the MUPE platform we have been able to concentrate on the applicatins themselves, not enablers for applications.

5.11 Contribution and research questions in papers

In this dissertation, we have dealt with the following research questions:

1. How to create and analyse the User Interface for a locationlbased application?
2. How to rapidly develop multi-user location-based services for mobile phones, and what kind of infrastructure is needed for this?
3. How to support user-creation of mobile location-based content?
4. What are the characteristics of location-based applications?

The first question is dealt in papers (i-v), which contained location-based navigation applications in wearable computers. The paper (i) offers an analytical tool for analysing location-based UIs. The papers (ii-iv) present different UIs: (ii) a minimal screen real estate consuming HMD UI, (iii) 2D map UIs, (iv) a 3D map UI. Paper (v) presents an extension to the WalkMap platform, which allows usability testing of any software in the wearable computer.

The second research question is dealt with in papers (vi-viii). Paper (vi) discusses what are the needs of the end-user for designing such a system. Paper (vii) presents MUPE, which is a platform supporting multi-user context-aware applications, featuring many useful features for rapid development. Paper (viii) presents a location extension for the MUPE platform, which allows rapid development of location-based services.

The high-end wearable platform used in papers (i-v) raised several needs, one of which was apparent after developing the usability test software (v), one of the most needed feature of the system would be to share information between users. The third research question builds on top of the wearable research, and implements end user created content into the mobile phone platform MUPE. Paper (vi) discusses the user requirements for user-created content. The platform and games presented in (vii) allow the users to create their own personal avatar, and communicate with each other using text. Paper (viii) further refines this idea by allowing the players of games to create the structure of the virtual world, by exploring the real world. The users do not need to be aware of the fact that they are constantly creating content into the virtual world.

The final research question is the big picture, that is, what are the characteristics of location-based applications. This is dealt throughout this dissertation, as depicted in Figure 1.1 (page 6). The four building blocks we have presented in this dissertation: Positioning, Landmarks, Application Logic, and User Interface, can all be implemented differently either in a standalone device, or distributed in the network. In our opinion, they are the defining characteristics of a location-based application, and all kinds of combinations are needed. Positioning on the user terminal offers reliability as it is always carried with the user, but with the cost of added power consumption. The landmarks in the user terminal makes sure they are always available, although they might not be up-to-date. If the application logic is in a single device, it is always available, but also always consumes power when active. Finally, there are many ways in which the UI can be created, and the application at hand defines what is the most suitable one.

Chapter 6

Conclusions and Discussion

In this dissertation, we have presented our work on developing new techniques for location-based applications. The three main areas covered in this work are: **Context- and location-awareness**, which is the backbone for the application development. **Combining the real and virtual worlds**, that is, what kinds of UIs can be created into the virtual world, and how to combine the real and virtual worlds to create location-based applications and UIs. **The locations-based applications themselves**, and technologies and structure needed for the development. We have presented research in each area with a thorough literature review, and also highlighted our contribution in each area. Our work includes techniques in each of the three areas, and we have created a theoretical analysis method, new user interface techniques, and an application platform that enables location-based applications in mobile phones.

In this work, we have covered the following research questions:

1. How to create and analyse the User Interface for a location-based application?

2. How to rapidly develop multi-user location-based services for mobile phones, and what kind of infrastructure is needed for this?

3. How to support user-creation of mobile location-based content?

4. What are the characteristics of location-based applications?

We present a series of papers that covered these issues. The papers can be divided to two tracks. The first track uses a wearable computer for developing high-end location-based applications. Such applications are complex, and also require good input and output device capabilities. The second track uses standard devices, namely mobile phones, to develop location-based applications that can be used with standard everyday devices.

The first research questions is dealt throughout our papers. The second and third research questions were raised by the wearable research, where too much focus was spent on the hardware platform. A wearable hardware platform becomes an essential part of research, eventhough our focus was on application development. Also, the application experience should be influenced by the users themselves, and the entire experience should be customisable by the multiple users in it.

The final research questions ties all our papers together, and it tries to find the main building blocks of location-based applications. In this dissertation, we developed four main building blocks for location-based applications that are:

1. *Positioning system* is the basis for everything. The position or location of a device, user or object needs to be known to build location-based applications, and there are many possible technologies to choose from.

2. *Location-based data* is the collection of objects or landmarks that is used in the application. This data can be personal, or shared among many users, and it can be stored locally, or in a network server.

3. *The application logic* determines how the data is used. A location-based application can be a map service, a game, or a friend finder. The application logic can reside in a single device only, or it can be distributed.

4. *The Human-Computer Interaction* defines the User Interface (UI) and how the users access the systems. Since location-based applications cover a very wide range of applications, there is a lot of variance in the UIs. AR applications overlay data directly on top of reality, whereas Short Message Service (SMS) based services only present a list of local objects.

Each of these building blocks can reside in the user's terminal, or they can be distributed into the network. How to implement each block is a deciding factor when developing location-based applications. A single device system is most likely to be always operable, as everything is carried in the single device, and no external parties are involved. On the other hand, sharing information among different applications requires some form of network connection. This dissertation offers new ways to analyse location-based application both in the form of the four building blocks, and in the taxonomy presented in research paper (i). Each research paper also offers new ways to develop applications and UIs.

Constructive research methods were applied in every paper, even the theretical papers were based on a lot of constructive research. Empirical methods were applied in the application evaluations in context compass, and WalkMap. Theoretical analysis was applied on the taxonomy paper, and the design of AR for casual user.

The future of this research goes in the direction of standard context- and location-based applications. MUPE offers a rapid development platform for context-aware applications in mobile phones and thus they have a large number of potential users. We aim to explore applications in user's own device, which allows us to focus on the applications, and not the devices. This also enables us to study the location-based applications in a large scale, in the potential end-users devices. It is also possible, that the classification methods proposed in this dissertation do not cover all possible location-aware applications, and the more applications are made, the more information we will get regarding this.

MUPE already offers many context-aware features, like Bluetooth for social context, client-side location-awareness with an external GPS module, and a camera module for 2D visual tags. Public displays in the environment can be connected to MUPE applications, which enables new public information channels for MUPE [93]. There are constantly new and interesting context-aware technologies emerging, which should be incorporated to the platform.

In the future, MUPE will be developed together with the open source community around it to better support rapid application development. As this dissertation pointed out, location-based applications could exist widely, as all technologies are mature, but not many widely used applications exist. That was one of the key design drivers behind MUPE, to speed up the application development and see user generated location-aware applications in everyday use. The themes used in the wearable applications will make their way into MUPE applications as technology matures. AR techniques are ideal when dealing with the real world, and already we are seeing the first AR applications in mobile phones.

Since the beginning of 2006 MUPE has moved towards its goals more rapidly with the help of Tekes (the Finnish Funding Agency for Technology and Innovation). TEKES currently funds a large development project *MoMUPE* where MUPE is developed by us in NRC (Nokia Research Center), TUT (Tampere University of Technology), LUT (Lappeenranta University of Technology), HIIT (Helsinki Institute of Information Technology), and VTT (Technical Research Centre of Finland). Our aim is to make context-aware applications in mobile phones widely used. To achieve this goal, we are developing a lot of context-aware games and playful applications that can be used in any mobile phone with J2ME.

Our current focus is on games and playful applications for a simple reason as pointed out in this dissertation. Sensors might not always provide accurate or correct context-aware information, and many applications (for example security and navigation) rely on accurate information. Games are much more forgiving, as a false input just generates a different state in the game. Because of the tolerance of inaccurate information, games will be the driving force of context-awareness in our opinion. Games are played by millions of people, and the games can be improved by making a link to the real world. If this one area can drive context-awareness forward, it is possible that the applications can become a natural part of our daily lives.

References

[1] Jsr-179 expert group. location api for java 2 micro edition. Technical report, Sun, 2003.

[2] 3GPP. Difference (e-otd) and global positioning system (gps) positioning methods (release 1999). Technical report, 3GPP, 1999.

[3] G. D. Abowd, C. G. Atkeson, J. Hong, S. Long, R. Kooper, and M. Pinkerton. Cyberguide: A mobile context-aware tour guide. *Wireless Networks*, 3(5):421 – 433, Oct 1997.

[4] G. D. Abowd and E. D. Mynatt. Charting past, present, and future research in ubiquitous computing. *ACM Trans. Comput.-Hum. Interact.*, 7(1):29–58, 2000.

[5] B. Avery, B. H. Thomas, J. Velikovsky, and W. Piekarski. Outdoor augmented reality gaming on five dollars a day. In *CRPIT '40: Proceedings of the Sixth Australasian conference on User interface*, pages 79–88, Darlinghurst, Australia, Australia, 2005. Australian Computer Society, Inc.

[6] R. Azuma, Y. Baillot, R. Behringer, S. Feiner, S. Julier, and B. MacIntyre. Recent advances in augmented reality. *IEEE Computer Graphics and Applications*, 21(6):34–47, 2001.

[7] P. Bahl and V. Padmanabhan. Radar: an in-building rf-based user location and tracking system. In *INFOCOM 2000. Nineteenth Annual Joint Conference of the IEEE Computer and Communications Societies.*, volume 2, pages 775–784. IEEE, 2000.

[8] F. Ballesteros, G. Guardiola, E. Soriano, and K. Leal. Traditional systems can work well for pervasive applications. a case study: Plan 9 from bell labs becomes ubiquitous. In *Third IEEE International Conference on Pervasive Computing and Communications, 2005. PerCom 2005*, pages 295 – 299. IEEE, 2005.

[9] J. Bardram. The java context awareness framework (jcaf) - a service infrastructure and programming framework for context-aware applications. In H. Gellersen, R. Want, and A. Schmidt, editors, *Third International Conference, PERVASIVE 2005*, volume 3468 of *Proceedings Series: Lecture Notes in Computer Science*, pages 98–115. Springer, May 2005.

[10] J. E. Bardram. Applications of context-aware computing in hospital work: examples and design principles. In *SAC '04: Proceedings of the 2004 ACM symposium on Applied computing*, pages 1574–1579, New York, NY, USA, 2004. ACM Press.

[11] M. Bauer, T. Heiber, G. Kortuem, and Z. Segall. A collaborative wearable system with remote sensing. In *ISWC '98: Proceedings of the 2nd IEEE International Symposium on Wearable Computers*, pages 10–17. IEEE Computer Society, 1998.

[12] S. Benford. Future location-based experiences. Technical report, JISC, 2005.

[13] M. Billinghurst, A. Cheok, S. Prince, and H. Kato. Real world teleconferencing. *IEEE Comput. Graph. Appl.*, 22(6):11–13, 2002.

[14] M. Billinghurst and H. Kato. Collaborative augmented reality. *Commun. ACM*, 45(7):64–70, 2002.

[15] M. Billinghurst, H. Kato, S. Bee, and J. Bowskill. Asymmetries in collaborative wearable interfaces. In *ISWC '99: Proceedings of the 3rd IEEE International Symposium on Wearable Computers*, page 133. IEEE Computer Society, 1999.

[16] M. Billinghurst, H. Kato, and I. Poupyrev. The magicbook: Moving seamlessly between reality and virtuality. *IEEE Comput. Graph. Appl.*, 21(3):6–8, 2001.

[17] S. Björk, 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.

[18] A. Blythe, C. Wright, and F. Monk. Little brother: could and should wearable computing technologies be applied to reducing older people's fear of crime? *Personal Ubiquitous Comput.*, 8(6):402–415, 2004.

[19] A. Burak and T. Sharon. Usage patterns of friendzone: mobile location-based community services. In *MUM '04: Proceedings of the 3rd international conference on Mobile and ubiquitous multimedia*, pages 93–100, New York, NY, USA, 2004. ACM Press.

[20] H. Chen, T. Finin, and A. Joshi. An ontology for context-aware pervasive computing environments. *Knowl. Eng. Rev.*, 18(3):197–207, 2003.

[21] K. Cheverst, N. Davies, K. Mitchell, A. Friday, and C. Efstratiou. Developing a context-aware electronic tourist guide: some issues and experiences. In *CHI '00: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 17–24, New York, NY, USA, 2000. ACM Press.

[22] R. DeVaul, M. Sung, J. Gips, and A. S. Pentland. Mithril 2003: Applications and architecture. In *ISWC '03: Proceedings of the 7th IEEE International Symposium on Wearable Computers*, page 4. IEEE Computer Society, 2003.

[23] R. W. DeVaul, A. S. Pentland, and V. R. Corey. The memory glasses: Subliminal vs. overt memory support with imperfect information. In *ISWC '03: Proceedings of the 7th IEEE International Symposium on Wearable Computers*, page 146, Washington, DC, USA, 2003. IEEE Computer Society.

[24] A. K. Dey. *Providing Architectural Support for Building Context-Aware Applications*. PhD thesis, College of Computing, Georgia Institute of Technology, 2000.

[25] A. K. Dey and G. D. Abowd. Toward a better understanding of context and context-awareness. Technical report, GVU Technical Report GIT-GVU-99-22, College of Computing, Georgia Institute of Technology, 1999.

[26] A. K. Dey, D. Salber, G. D. Abowd, and M. Futakawa. The conference assistant: Combining context-awareness with wearable computing. In *ISWC '99: Proceedings of the 3rd IEEE International Symposium on Wearable Computers*, page 21. IEEE Computer Society, 1999.

[27] G. M. Djuknic and R. E.Richton. Geolocation and assisted gps. *Computer*, 34(2):123–125, Feb 2001.

[28] N. Eagle and A. Pentland. Social serendipity: Mobilizing social software. *IEEE Pervasive Computing*, 4(2):28–34, April-June 2005.

[29] S. Ertan, C. Lee, A. Willets, H. Tan, and A. Pentland. A wearable haptic navigation guidance system. In *ISWC '98: Proceedings of the 2nd IEEE International Symposium on Wearable Computers*, page 164. IEEE Computer Society, 1998.

[30] EUROCONTROL and IfEN. Wgs 84 implementation manual. Technical report, EUROCONTROL AND IfEN, 1998.

[31] J. Farringdon, A. J. Moore, N. Tilbury, J. Church, and P. D. Biemond. Wearable sensor badge and sensor jacket for context awareness. In *ISWC '99: Proceedings of the 3rd IEEE International Symposium on Wearable Computers*, page 107, Washington, DC, USA, 1999. IEEE Computer Society.

[32] S. Feiner, B. MacIntyre, T. Hollerer, and A. Webster. A touring machine: Prototyping 3d mobile augmented reality systems for exploring the urban environment. In *ISWC '97: Proceedings of the 1st IEEE International Symposium on Wearable Computers*, page 74. IEEE Computer Society, 1997.

[33] S. S. Fisher, M. McGreevy, J. Humphries, and W. Robinett. Virtual environment display system. In *SI3D '86: Proceedings of the 1986 workshop on Interactive 3D graphics*, pages 77–87, New York, NY, USA, 1987. ACM Press.

[34] M. Flintham, R. Anastasi, S. Benford, A. Drozd, J. Mathrick, D. Rowland, A. Oldroyd, J. Sutton, N. Tandavanitj, M. Adams, and J. Row-Farr. Uncle roy all around you: mixing games and theatre on the city streets. In C. Marinka and R. Joost, editors, *Level Up Conference Proceedings*, Utrecht, November 2003. University of Utrecht.

[35] M. Flintham, S. Benford, R. Anastasi, T. Hemmings, A. Crabtree, C. Greenhalgh, N. Tandavanitj, M. Adams, and J. Row-Farr. Where on-line meets on the streets: experiences with mobile mixed reality games. In *CHI '03: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 569–576, New York, NY, USA, 2003. ACM Press.

[36] C. D. Flora, O. Riva, K. Raatikainen, and S. Russo. Supporting mobile context-aware applications through a modular service infrastructure. In *Poster Proceedings of 6th International UbiComp 2004 Conference*, September 2004.

[37] C. Geiger, B. Kleinnjohann, C. Reimann, and D. Stichling. Mobile ar4all. In *ISAR '01: Proceedings of the IEEE and ACM International Symposium on Augmented Reality (ISAR'01)*, page 181. IEEE Computer Society, 2001.

[38] H. W. Gellersen, A. Schmidt, and M. Beigl. Multi-sensor context-awareness in mobile devices and smart artifacts. *Mob. Netw. Appl.*, 7(5):341–351, 2002.

[39] S. Greenberg 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.

[40] C. Greenhalgh and S. Benford. Massive: a collaborative virtual environment for teleconferencing. *ACM Trans. Comput.-Hum. Interact.*, 2(3):239–261, 1995.

[41] R. Guarneri, A. Sollund, D. Marston, E. Fossbak, B. Berntsen, G. Nygreen, G. Gylterud, R. Bars, and A. Kerdraon. Report of state of the art in personalisation. common framework. Technical report, IST Project Nř 506775, February 2004.

[42] J. Hightower and G. Borriello. Location systems for ubiquitous computing. *Computer*, 34(8):57–66, 2001.

[43] F. Hohl, U. Kubach, A. Leonhardi, K. Rothermel, and M. Schwehm. Next century challenges: Nexus - an open global infrastructure for spatial-aware applications. In *MobiCom '99: Proceedings of the 5th annual ACM/IEEE international conference on Mobile computing and networking*, pages 249–255. ACM Press, 1999.

[44] R. Hull, P. Neaves, and J. Bedford-Roberts. Towards situated computing. In *ISWC '97: Proceedings of the 1st IEEE International Symposium on Wearable Computers*, page 146. IEEE Computer Society, 1997.

[45] H. Ishii and B. Ullmer. Tangible bits: towards seamless interfaces between people, bits and atoms. In *CHI '97: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 234–241, New York, NY, USA, 1997. ACM Press.

[46] E. Kaasinen. *User acceptance of mobile services - value, ease of use, trust and ease of adoption.* PhD thesis, Tampere University of Technology,, June 2005.

[47] K. J. Kangas and J. Röning. Using mobile code to create ubiquitous augmented reality. *Wirel. Netw.*, 8(2/3):199–211, 2002.

[48] J. Kolari, T. Laakko, T. Hiltunen, V. Ikonen, M. Kulju, R. Suihkonen, S. T. T., and Virtanen. *Context-Aware Services for Mobile Users (Technology and User Experiences)*, volume 539 of *VTT Publications*. VTT, 2004.

[49] M. Korkea-aho. Context-aware applications survey. Technical report, Department of Computer Science, Helsinki University of Technology, April 2000.

[50] P. Korpipää, M. Koskinen, J. Peltola, S.-M. Mäkelä, and T. Seppänen. Bayesian approach to sensor-based context awareness. *Personal Ubiquitous Computing*, 7(2):113–124, 2003.

[51] P. Korpipää and J. Mäntyjärvi. An ontology for mobile device sensor-based context awareness. In *Proceedings of the 4th International and Interdisciplinary Conference on Modeling and Using Context*, volume 2680, pages 451–459. LNAI, Springer-Verlag, 2003.

[52] P. Korpipää, J. Mäntyjärvi, J. Kela, H. Keränen, and E.-J. Malm. Managing context information in mobile devices. *IEEE Pervasive Computing*, 2(3):42–51, July-September 2003.

[53] G. Kortuem, M. Bauer, and Z. Segall. Netman: the design of a collaborative wearable computer system. *Mob. Netw. Appl.*, 4(1):49–58, 1999.

[54] H. Lakkala. Context exchange protocol specification. Technical report, Nokia, 2003.

[55] M. Lampe, M. Strassner, and E. Fleisch. A ubiquitous computing environment for aircraft maintenance. In *SAC '04: Proceedings of the 2004 ACM symposium on Applied computing*, pages 1586–1592, New York, NY, USA, 2004. ACM Press.

[56] S.-W. Lee and K. Mase. Activity and location recognition using wearable sensors. *IEEE Pervasive Computing*, 1(3):24–32, 2002.

[57] J. Lehikoinen. *Interacting with Wearable Computers: Techniques and Their Application in Wayfinding Using Digital Maps*, volume 207 of *Acta Electronica Universitatis Tamperensis.* University of Tampere, 2002.

[58] J. Lehikoinen, J. Holopainen, M. Salmimaa, and A. Aldrovandi. Mex: A distributed software architecture for wearable computers. In *ISWC '99: Proceedings of the 3rd IEEE International Symposium on Wearable Computers*, page 52. IEEE Computer Society, 1999.

[59] J. Lehikoinen and M. Röykkee. N-fingers: A finger-based interaction technique for wearable computers. *Interacting with Computers*, 13:601–625, 2001.

[60] J. Lehikoinen and R. Suomela. Accessing context in wearable computers. *Personal Ubiquitous Computing*, 6(1):64–74, 2002.

[61] J. Lehikoinen and R. Suomela. Perspective map. In *Proceedings of the Sixth International Symposium on Wearable Computers, 2002. (ISWC 2002)*, pages 171– 178, 2002.

[62] S. Long, R. Kooper, G. D. Abowd, and C. G. Atkeson. Rapid prototyping of mobile context-aware applications: the cyberguide case study. In *MobiCom '96: Proceedings of the 2nd annual international conference on Mobile computing and networking*, pages 97–107. ACM Press, 1996.

[63] K. Lynch. *The Image of the City.* The MIT Press. VTT, 1960.

[64] B. MacIntyre, M. Gandy, S. Dow, and J. D. Bolter. Dart: a toolkit for rapid design exploration of augmented reality experiences. In *UIST '04: Proceedings of the 17th annual ACM symposium on User interface software and technology*, pages 197–206. ACM Press, 2004.

[65] S. Mann. An historical account of the 'wearcomp' and 'wearcam' inventions developed for applications in 'personal imaging'. In *ISWC '97: Proceedings of the 1st IEEE International Symposium on Wearable Computers*, page 66. IEEE Computer Society, 1997.

[66] S. Mann. "sousveillance": inverse surveillance in multimedia imaging. In *MULTIMEDIA '04: Proceedings of the 12th annual ACM international conference on Multimedia*, pages 620–627, New York, NY, USA, 2004. ACM Press.

[67] J. Mäntyjärvi. *Sensor-based Context Recognition for Mobile Applications*. PhD thesis, University of Oulu, December 2003.

[68] J. Mäntyjärvi, J. Himberg, and P. Huuskonen. Collaborative context recognition for handheld devices. In *PERCOM '03: Proceedings of the First IEEE International Conference on Pervasive Computing and Communications*, pages 161–168. IEEE Computer Society, 2003.

[69] P. Milgram and F. Kishino. A taxonomy of mixed reality visual displays. *IEICE Transactions on Information Systems*, E77-D(12):1321–1329, December 1994.

[70] G. E. Moore. Cramming more components onto integrated circuits. *Electronics*, 38(8), April 1965.

[71] M. J. Pascoe. Adding generic contextual capabilities to wearable computers. In *ISWC '98: Proceedings of the 2nd IEEE International Symposium on Wearable Computers*, pages 92–99. IEEE Computer Society, 1998.

[72] J. Patten, H. Ishii, J. Hines, and G. Pangaro. Sensetable: a wireless object tracking platform for tangible user interfaces. In *CHI '01: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 253–260, New York, NY, USA, 2001. ACM Press.

[73] A. Pentland. Healthwear: medical technology becomes wearable. *Computer*, 37(5):42–49, May 2004.

[74] A. Pentland. Socially aware, computation and communication. *Computer*, 38(3):33–40, March 2005.

[75] W. Piekarski, B. Avery, B. H. Thomas, and P. Malbezin. Integrated head and hand tracking for indoor and outdoor augmented reality. In *VR '04: Proceedings of the IEEE Virtual Reality 2004 (VR'04)*, page 11, Washington, DC, USA, 2004. IEEE Computer Society.

[76] W. Piekarski and B. H. Thomas. Interactive augmented reality techniques for construction at a distance of 3d geometry. In *EGVE '03: Proceedings of the workshop on Virtual environments 2003*, pages 19–28. ACM Press, 2003.

[77] K. Raatikainen, H. B. Christensen, and T. Nakajima. Application requirements for middleware for mobile and pervasive systems. *SIGMOBILE Mob. Comput. Commun. Rev.*, 6(4):16–24, 2002.

[78] I. Rakkolainen, S. Pulkkinen, and A. Heinonen. Visualizing real-time gps data with internet's vrml worlds. In *GIS '98: Proceedings of the 6th ACM international symposium on Advances in geographic information systems*, pages 52–56, New York, NY, USA, 1998. ACM Press.

[79] G. Reitmayr and D. Schmalstieg. Location based applications for mobile augmented reality. In *CRPITS '18: Proceedings of the Fourth Australian user interface conference on User interfaces 2003*, pages 65–73. Australian Computer Society, Inc., 2003.

[80] J. Rekimoto, Y. Ayatsuka, and K. Hayashi. Augment-able reality: Situated communication through physical and digital spaces. In *ISWC '98: Proceedings of the 2nd IEEE International Symposium on Wearable Computers*, page 68. IEEE Computer Society, 1998.

[81] B. J. Rhodes. The wearable remembrance agent: a system for augmented memory. In *ISWC '97: Proceedings of the 1st IEEE International Symposium on Wearable Computers*, page 123. IEEE Computer Society, 1997.

[82] M. Rohs. Real-world interaction with camera-phones. In *2nd International Symposium on Ubiquitous Computing Systems (UCS 2004)*, pages 39–48, Tokyo, Japan, Nov. 2004.

[83] D. Salber, 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.

[84] N. Sawhney and C. Schmandt. Nomadic radio: Speech & audio interaction for contextual messaging in nomadic environments. *ACM Transactions on Computer Huma n Interaction*, 7(3):353–383, 2000.

[85] W. N. Schilit. *A system architecture for context-aware mobile computing.* PhD thesis, Columbia Univ., New York, NY, 1995.

[86] D. Schmalstieg, A. Fuhrmann, G. Hesina, Z. Szalavári, L. M. E. ao, M. Gervautz, and W. Purgathofer. The studierstube augmented reality project. *Presence: Teleoper. Virtual Environ.*, 11(1):33–54, 2002.

[87] J. Scott and B. Dragovic. Audio location: Accurate low-cost location sensing. In H. Gellersen, R. Want, and A. Schmidt, editors, *Third*

International Conference, PERVASIVE 2005, volume 3468 of *Proceedings Series: Lecture Notes in Computer Science*, pages 1–18. Springer, May 2005.

[88] I. O. Sebe, J. Hu, S. You, and U. Neumann. 3d video surveillance with augmented virtual environments. In *IWVS '03: First ACM SIGMM international workshop on Video surveillance*, pages 107–112, New York, NY, USA, 2003. ACM Press.

[89] R. Smith, W. Piekarski, and G. Wigley. Hand tracking for low powered mobile ar user interfaces. In *6th Australasian User Interface Conference, Newcastle, NSW*, Jan 2005.

[90] O. Sotamaa. All the world's a botfighter stage: Notes on location-based multi-player gaming. In *Proceedings of the Computer Games and Digital Cultures Conference*, pages 35–44. Tampere University Press, 2002.

[91] T. Starner. The challenges of wearable computing: Part 1. *IEEE Micro*, 21(4):44–52, July/August 2001.

[92] R. Suomela. A method and apparatus to browse and access downloaded contextual information. Patent application (wo 01/88687), World Intellectual Property Organisation, 2001.

[93] R. Suomela. Public screens and private mobile phone screens in multi-player games. In *The Workshop on Applications for Wireless Communications WAWC*, volume 211 of *Acta Universitatis Lappeenrantaensis*, pages 17–26, 2005.

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

[95] B. Thomas, B. Close, J. Donoghue, J. Squires, P. de Bondi, M. Morris, and W. Piekarski. Arquake: An outdoor/indoor augmented reality first person application. In *ISWC '00: Proceedings of the 4th IEEE International Symposium on Wearable Computers*, page 139. IEEE Computer Society, 2000.

[96] B. Thomas, V. Demczuk, W. Piekarski, D. Hepworth, and B. Gunther. A wearable computer system with augmented reality to support terrestrial navigation. In *Second International Symposium on Wearable Computers*, pages 168–171, Oct 1998.

[97] K. Toyama, R. Logan, and A. Roseway. Geographic location tags on digital images. In *MULTIMEDIA '03: Proceedings of the eleventh ACM international conference on Multimedia*, pages 156–166, New York, NY, USA, 2003. ACM Press.

[98] B. Ullmer, H. Ishii, and R. J. K. Jacob. Token+constraint systems for tangible interaction with digital information. *ACM Trans. Comput.-Hum. Interact.*, 12(1):81–118, 2005.

[99] D. Wagner, T. Pintaric, F. Ledermann, and D. Schmalstieg. Towards massively multi-user augmented reality on handheld devices. In H. Gellersen, R. Want, and A. Schmidt, editors, *Third International Conference, PERVASIVE 2005*, volume 3468 of *Proceedings Series: Lecture Notes in Computer Science*, pages 208–219. Springer, May 2005.

[100] D. Wagner and D. Schmalstieg. Artoolkit on the pocketpc platform. In *IEEE International Workshop on Augmented Reality Toolkit*, number 7, pages 14–15, October 2003.

[101] R. Want, A. Hopper, V. Falcao, and J. Gibbons. The active badge location system. *ACM Trans. Inf. Syst.*, 10(1):91–102, 1992.

[102] R. Want, B. N. Schilit, N. I. Adams, R. Gold, K. Petersen, D. Goldberg, J. R. Ellis, and M. Weiser. An overview of the PARCTAB ubiquitous computing experiment. *IEEE Personal Communications*, 2(6):28–43, Dec 1995.

[103] M. Weiser. The computer for the twenty-first century. *Scientific American*, 265(3):94–104, 1991.

[104] N. Weißenberg, A. Voisard, and R. Gartmann. Using ontologies in personalized mobile applications. In *GIS '04: Proceedings of the 12th annual ACM international workshop on Geographic information systems*, pages 2–11. ACM Press, 2004.

[105] J. Yang, W. Yang, M. Denecke, and A. Waibel. Smart sight: A tourist assistant system. In *ISWC '99: Proceedings of the 3rd IEEE International Symposium on Wearable Computers*, page 73. IEEE Computer Society, 1999.

[106] J.-D. Yim 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.

[107] Ekahau, Inc. Ekahau Positioning. http://www.ekahau.com/.

[108] Groundspeak. Geocaching. http://www.geocaching.com/.

[109] Lambda MOO. http://www.moo.mud.org/.

[110] It's Alive. BotFighters. Commercial Mobile Game, 2001.

[111] Newt Games. Mogi. Commercial Mobile Game, 2003.

[112] AR Toolkit. http://www.hitl.washington.edu/artoolkit/, 2005.

[113] Benefon Oyj. Company website. http://www.benefon.com/products/index.htm, 2005.

[114] Enhanced 911 - Wireless Services. http://www.fcc.gov/911/enhanced/, 2005.

[115] Garmin International Inc. eTrex Vista, 2005.

[116] GeoVector Corp. Company website. http://www.geovector.com/, 2005.

[117] Google. Google Earth. A multiplatform navigation tool, 2005.

[118] GPS Pilot S.A.R.L. GPS Pilot. http://www.gpspilot.com/Products.htm, 2005.

[119] Handykey Corporation. Twiddler2, 2005.

[120] International Organization for Standardisation. Open systems interconnection (OSI). , 2005.

[121] NASA Human-Centered Systems Lab. Human-Centered Systems Lab, 2005.

[122] Nokia. Nokia Wireless GPS Module LD-1W. http://www.nokia.com/, 2005.

[123] The Lego Group. Lego Mindstorms. Internet, 2005.

[124] Tinmith website. http://www.tinmith.net/, 2005.

[125] Vindigo, Inc. MapQuest Mobile. http://www.mapquest.com/mobile/main.adp, 2005.

[126] Xybernaut Corporation. Xybernaut wearable and mobile computing. http://www.xybernaut.com/, 2005.

Chapter 7

Publications

Publication 1: Taxonomy for visualizing location-based information.

R. Suomela, J. Lehikoinen. Taxonomy for visualizing location-based information. *Virtual Reality, vol. 6, no. 2*, 71-82.

Publication 2: Context Compass.

R. Suomela, J. Lehikoinen. Context Compass. *Proceedings of the Fourth International Symposium on Wearable Computers* , 147-154.

Publication 3: WalkMap Developing an Augmented Reality Map Application for Wearable Computers.

J. Lehikoinen, R. Suomela. WalkMap Developing an Augmented Reality Map Application for Wearable Computers. *Virtual Reality, vol. 6, no. 1* , 33 - 44.

Publication 4: The evolution of perspective view in WalkMap.

R. Suomela, K. Roimela, J. Lehikoinen. The evolution of perspective view in WalkMap. *Personal and Ubiquitous Computing, vol. 7, no. 5*, 249-262, 2003.

Publication 5: A system for evaluating augmented reality user interfaces in wearable computers.

R. Suomela, J. Lehikoinen, I. Salminen. A system for evaluating augmented reality user interfaces in wearable computers. *Proceedings of the Fifth International Symposium on Wearable Computers (ISWC), 2001, 77-84. ©2001 IEEE Computer Society.*

Publication 6: Augmented Reality for a Casual User: Designing Tools for Interaction with the Virtual World.

R. Suomela, J. Mattila, E. Räsänen, T. Koskinen. Augmented Reality for a Casual User: Designing Tools for Interaction with the Virtual World. Proceedings of the Computer Games and Digital Cultures Conference *, 185-190.*

Publication 7: Open-Source Game Development with the Multi-User Publishing Environment (MUPE) Application Platform.

R. Suomela, 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.*

Publication 8: Rapid Prototyping of Location-Based Games with the Multi-User publishing Environment Application Platform.

R. Suomela, 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.*

Tampereen teknillinen yliopisto
PL 527
33101 Tampere

Tampere University of Technology
P.O. Box 527
FIN-33101 Tampere, Finland