CLOUD COMPUTING AND CONTEXT-AWARENESS – A STUDY OF THE ADAPTED USER EXPERIENCE

A thesis submitted for the degree of Doctor of Philosophy

by

Tor-Morten Grønli

School of Information System, Computing and Mathematics, Brunel University

June 2012

ABSTRACT

Today, mobile technology is part of everyday life and activities and the mobile ecosystems are blossoming, with smartphones and tablets being the major growth drivers. The mobile phones are no longer just another device, we rely on their capabilities in work and in private. We look to our mobile phones for timely and updated information and we rely on this being provided any time of any day at any place. Nevertheless, no matter how much you trust and love your mobile phone the quality of the information and the user experience is directly associated with the sources and presentation of information.

In this perspective, our activities, interactions and preferences help shape the quality of service, content and products we use. Context-aware systems use such information about end-users as input mechanisms for producing applications based on mobile, location, social, cloud and customized content services. This represents new possibilities for extracting aggregated user-centric information and includes novel sources for context-aware applications. Accordingly, a Design Research based approach has been taken to further investigate the creation, presentation and tailoring of user-centric information. Through user evaluated experiments findings show how multi-dimensional context-aware information can be used to create adaptive solutions tailoring the user experience to the users' needs. Research findings in this work; highlight possible architectures for integration of cloud computing services in a heterogeneous mobile environment in future context-aware solutions. When it comes to combining context-aware results from local computations with those of cloud based services, the results provide findings that give users tailored and adapted experiences based on the collective efforts of the two.

ACKNOWLEDGEMENTS

I would like to especially thank my supervisor Dr Gheorghita Ghinea who has offered invaluable assistance, support and feedback. Your knowledge and guidance made it possible to pursue and complete this research successfully.

I would also like to thank my colleagues at the Norwegian School of Information Technology and in particular give my genuine thanks to Bjørn J. Hanssen, Dean of The Norwegian School of Information Technology. Your encouragements, fruitful discussions and constant support made it possible to finish this work.

Additionally, I am thankful for the support and work provided by my colleague, Jarle Hansen, who helped me improve and expand on my own research.

And last but not least I owe my loving thanks to my wife Ragnhild L. Grønli and our children Vemund, Tuva and Håvard. Without your encouragement and understanding it would have been impossible for me to finish this research.

- Tor-Morten Grønli, 15/4 – 2012

LIST OF PUBLICATIONS

The following papers have been published (or have been submitted for publication) as a result of the research discussed in this thesis:

JOURNALS

1. Grønli, T.M. and Ghinea, G. (2010) "Three-dimensional context-aware tailoring of information". In *Online information review,* volume 34, issue 6, pp. 892-906

2. Grønli, T.M., Hansen, J. and Ghinea, G. (2012) "Mobility, Context and Cloud – Exploring Integration Issues in a Meeting Room Scenario". In *Journal of Interactive Mobile Technologies*, volume 6, issue 2, pp. 13-22

3. Grønli, T.M., Hansen, J. and Ghinea (n.d.) "The Tailored User Experience: A Multi Dimensional and Cloud Integrated Context-Aware Phone" In *Online Information Review* [Under Review]

4. Hansen, J. Grønli, T.M. and Ghinea, G. (n.d.) "Towards Cloud to Device Push Messaging on Android: A Performance Comparison". In *Software: Practice and Experience*, Wiley [Under Review]

PEER REVIEWED CONFERENCES

1. Grønli, T.M. and Ghinea, G. (2009) "Web 2.0 integration in a context-aware mobile PIM application". ICITST. IEEE conference proceedings

2. Grønli, T.M., Hansen, J. and Ghinea, G. (2010) "Android vs Windows Mobile vs Java ME: a comparative study of mobile development environments" Proceedings of the 3rd International Conference on *PErvasive Technologies Related to Assistive Environments*. ACM

3. Grønli, T.M., Hansen, J. and Ghinea, G. (2010) "Android, Java ME and Windows Mobile Interplay: The Case of a Context-Aware Meeting Room". Proceedings of the *24th IEEE International Conference on Advanced Information Networking and Applications Workshops, (WAINA)*. IEEE Computer

4. Grønli, T.M., Hansen, J. and Ghinea, G. (2010) "A Context-Aware Meeting Room: Mobile Interaction and Collaboration using Android, Java ME and Windows Mobile" Proceedings of the *34th Annual IEEE Computer Software and Applications Conference Workshops (COMPSACW)*. IEEE Computer

5. Grønli, T.M., Hansen, J. and Ghinea, G. (2011) "Integrated Context-Aware and Cloud-Based Adaptive Home Screens for Android Phones" Proceedings of the *Human-Computer Interaction: Interaction Techniques and Environments*. Springer

6. Grønli, T.M., Hansen, J. and Ghinea, G. (2011) "A Cloud on the Horizon: The Challenge of Developing Applications for Android and iPhone" Proceedings of the 4th *PErvasive Technologies Related to Assistive Environments*. ACM

7. Hansen, J. Grønli, T.M. and Ghinea, G. (2012) "Cloud to Device Push Messaging on Android: a Case Study". Proceedings of the *26th IEEE International Conference on Advanced Information Networking and Applications Workshops, (WAINA)*. IEEE Computer

BOOK CHAPTER

1. Grønli, T.M. and Bygstad, B. (2010) “Innovation in Mobile Services - A Design Research Approach.” *Handbook of Mobile Technology Research Methods*, edited by Steve Love. Nova Science Publishers, Inc.

RELATED PUBLICATIONS

1. Spyridonis, F., Grønli, T.M., Hansen, J. and Ghinea, G. (2012) “Evaluating the usability of a Virtual Reality-based Android application in managing the pain experience of wheelchair users”. In *34th Annual International Conference of the Engineering in Medicine and Biology Society,* 2012

2. Spyridonis, F., Hansen, J., Grønli, T.M and Ghinea, G. (n.d.), “PainDroid: An Android-based Virtual Reality Application for Pain Management”. In Transactions on Systems, Man, and Cybernetics-Part A, IEEE [Under Review]

TABLE OF CONTENTS

List of Figures

List of Tables

APPENDICES

Chapter 1

Introduction

Today, mobile technology is part of everyday life and activities. Everyone carries around a device, some having even more than one device at home. As such, the mobile ecosystem is blossoming, with smartphones and tablets being the major growth drivers of 2011 (Gartner Group, 2012a). In the global market, smartphones sales are increasing enormously, growing 62.7% from 2010 to 2011 and reaching a total of just below 500 million devices in 2011 in the US (Canalys, 2012). Worldwide sales of mobile devices also continue to increase and are reported to have reached 440.5 million units in the third quarter of 2011, up 5.6% from the same period in 2010. In this market the smartphones are the fastest growing group, and in the third quarter of 2011, these devices made up 26% of the world's total mobile phone sales (Gartner Group, 2011c). The total smartphone sale to end users reached 472 million units for the whole of 2011, which accounted for 31% of total mobile phone sales and up an overwhelming 58% from 2010 (Gartner Group, 2012b). Gartner Group expects the market to continue to increase for 2012 as well, although at somewhat slower pace with a prediction of a 7% increase for 2012. This indicates an enormous market to target, with corresponding opportunities for innovation and large impact of solutions.

The impressive number of sales, and future predictions, point in the direction of an ever expanding market and a continuous trend. One realizes that the mobile phone, and in particular the smartphone, is something completely different from the old fixed-line telephone. Your smartphone certainly is a computer, with an impressive range of functionalities, including full Internet access. It is also your bank, music player and chart plotter with GPS. It is on its way to becoming your future payment

mechanism for the metro, car parking and private sales. Solutions like those from iZettle (2012), Square (2012) and Visa (2012) are moving the mobile payment experience forward. However it is not only about business, it is also about something more emotional. The mobile device stays with you day and night, it increasingly contains much of your personal life, and it is something you really dread to lose. In short, it is your companion. A companion is someone that can be trusted, and knows your situation. A companion is sensitive to your needs and makes your daily life easier. In order to do so, the companion must know your current context, and be able to adapt to various contexts.

The world of mobile devices, services and ecosystems is a rapidly changing environment. Still some services, technologies and trends are more rigorous than others. The move towards a mobile-centric approach, away from traditional desktop with point and click, takes us towards the mobile environment with touch interfaces, gestures and voice/video recognition. These new interfaces represent a new paradigm and new interfaces facilitate for new user experiences and new interactions with objects in the environment. Our activities, interactions and preferences help shape the quality of service, content and products we use. Context-aware systems use such information about the end-user as input mechanisms for producing applications based on mobile, location, social and customized content services. It is not surprising, therefore, that Gartner Group (2011a) included context-aware services, mobile-centric applications and interfaces and cloud computing as three out of ten strategic technologies for 2012. The link to cloud applications is quite interesting and presents new possibilities for extracting aggregated user-centric information and includes this as new sources for context-aware applications.

Accordingly, this chapter is structured as follows: Section one starts off by describing mobile information technology. Further, since mobile information technology is expanding and integrating more and more new parts of our lives. Section two looks into ubiquitous computing and gives an overview of how technology seamless integrates in our world. Besides such integration of services and technology, content tailoring and information presentation are important subjects to capture the users' attention and present intuitive user experiences and timely

information; Section three therefore explores work pertaining to the area of context-aware computing, before conclusions are drawn in section four.

1.1 Mobile Communication

As expounded in the introduction, mobile computing has evolved quite a lot over the last 10 years. Smartphones have gone from being a device targeting business consultants to the public domain. The eco-system for mobile devices is changing rapidly, from the early days of feature phones with access to Internet browsing through the Wireless Application Protocol (WAP) pages, to todays feature-rich phones with on-device, browser built-in, web hypertext mark-up language version 5 (HTML 5) web solutions.

Cell phones emerged through the 1980s and became fairly common late that century, but almost exclusively as a device for placing calls. From mid 1990s the personal digital assistants (PDA) came to market. Even though the early PDAs were not mobile phones and able to make calls, they are still seen as a predecessor of today's smartphones. Moreover, as manufacturers started to improve essential hardware such as processor, memory and displays, the mobile phones shifted to become mobile devices with smart operating systems. Today, the cell phone has emerged into a full blown, small-scale computer, the modern smartphone. It is capable of making calls, taking pictures/video, connecting to online services, running games and utility applications, and can be used for payment and provide sensor information (Hall and Anderson, 2009).

Also for consumer market places a lot has changed. Apple started the race by launching their Apple App store. This revolutionized the mobile domain creating an immense amount of job opportunities and possibilities. Following Apple, Google launched its Android platform a year later, with its associated online shop for applications - the Google Market (recently rebranded to Google Play). The last successful launch is credited to Microsoft, who undertook a total make over of its mobile platform and re-launched it as Windows Phone in 2011. Together with this launch a new partnership with NOKIA emerged and the launch of the Microsoft application store, the Marketplace.

These new ecosystems create a whole range of new opportunities and possibilities for business innovation. Hardware vendors battle to stay on top of each other with the latest and best hardware specifications and software vendors participate in a commercial game advertising for the customers to put their money in their applications. In this chaotic world of devices, applications and manufacturers, it is increasingly important to create applications that enrich the user experience and deliver timely information and experiences. To see the overall perspective and remind us of the roots that started this, we need to look at the early days of mobile computing, in the origin of ubiquitous computing.

1.2 Computing Everywhere for Everyone

It is from the world of distributed systems in the mid 1970's and following in the early 1990's, that mobile computing and the field of ubiquitous computing emerged (Satyanarayanan, 2011). The concept of ubiquitous computing originated from the idea of making computer technology available through physical surroundings (Weiser, 1991) and make it available everywhere for everyone. The ideas and concepts first introduced and envisioned by Weiser (1993; Weiser, 1991) and his co-researchers at Xerox's Palo Alto Research Centre (PARC) are now over two decades old, but the development of mobile and ubiquitous computing has continued at impressive speed and changed the way we interact with a number of objects and the way we perform numerous tasks. In his articles, Weiser identified two crucially important issues, namely location and communication, which are necessary to initiate the "disappearance" of wired computing and create the conditions necessary for the birth and growth of location and context-aware computing (Weiser, 1991; Weiser, 1993; Weiser, 1994).

The ubiquitous computing paradigm is becoming a standard in our time as handheld devices (PDAs, mobile phones, tablets) are becoming cheaper and increasingly widespread. As time has passed since the original ideas of Weiser, several products have come publicly available and are more or less integrated in our daily life; Handheld devices, wearable computing and wireless LAN are all examples of such technology (Satyanarayanan, 2011). Hand in hand with ubiquitous computing the discipline of pervasive computing has also emerged. Together, these two paradigms

(further referred to as ubiquitous computing) have altered many processes of our world, ranging from production processes, to business and everyday life. Companies like IBM have offered a range of middleware services for mobile ubiquitous devices/computing with examples like operating systems, servers, security and personal information management. The applications and services available today from a whole range of manufacturers are numerous. As Kjeldskov and Paay (2006) stated, people's relations, ways of communication and coordination of social life have been changed by ubiquitous computing. Technologies like mobile devices, Short Messaging Service (SMS), online communities, video enabled communication are the driving forces behind this. Research over the years has looked at different trade-offs between integration, performance and application services. Developed applications, design pattern implementations and new interfaces are parts that shape the future technology. As mobile technology has emerged, it has been put to use in different kinds of businesses (Kurkovsky and Harihar, 2006). Still, new technology and advanced improvements are not enough to create user value. To be able to target the user in specific settings, we need to identify the current user sphere.

One such effort is the expansion of ubiquitous computing to our everyday world. Our cars are becoming connected, and often offer opportunities for wired or wireless connection and integration of devices such as our smartphones. Prototypes are produced from Ford, Toyota, Volvo and other vendors, on how smart devices can be used for extracting vehicle status information. The same goes for our homes where technology is becoming increasingly woven into the environment. Internet of things represents one part of the interconnected web of devices where estimates predict that by 2015 15 billion devices could be linked to the Internet, up from 7.5 billion currently (BBC News, 2011). Other environments getting connected are such as airplanes, city infrastructures and hospitals. Ubiquitous computing also takes diverse forms such as NFC chips for payment, *Google Goggles* application for seamless integration between images and product information or devices for golf ranges visualising the course ahead of the player (Economist, 2011). Although we are still in a wired world and have not yet come as far as described in Weiser's (1991) visions, technology has become considerably more accessible. The next step is to make the devices more aware of human behaviour and make them adapt. The first step in

adaptation is the ability to adapt to different situation and environments – contexts. As researchers gradually became interested in this, context-awareness emerged as a promising area of research.

1.3 Context-Aware Information

The initial ideas regarding ubiquitous computing and visible computing came from the research of Mark Weiser. Weiser (1991) identified location and communication as two vitally important issues to create technological solutions so unobtrusive that they weave themselves into the fabric of our everyday lives and, by this, facilitate the birth and growth of location and context-aware computing. "Recognizing that every situation has its own given context of information and events, this context will vary depending on the situation we are in, and the tasks we are about to perform" (Dourish, 2004). Several researchers have taken up the challenge of defining context and they all agree upon its value as a source of information. Generally speaking, context is often defined either to be time, location or action (Dey and Abowd, 1999; Dey, 2001; Schilit et al., 1994). The use of context in the design of applications is a possible solution to avoid vagueness and misinterpretation, if used correctly. Accordingly, developers and researchers see that the context is one important factor when designing new applications that open interesting possibilities for tailoring of applications for use in homes, at workplaces, on travel etc. One approach of merging the worlds of research and business is the one that is formed in context-aware computing. Dey and Abowd (1999) state that if we understand context fully in a given environment and setting, we would then be able to better choose what context-aware behaviours to sustain in applications. This could lead to more realistic applications and thereby applications more meaningful to the users. Edwards (2005) also exemplified this in his approach when he used context information in a layered fashion to build an adaptive application, and argued that context is a major part of our daily life and computing with support for sharing and using contextual information (context-aware computing) would improve the user interaction. When viewing people rather than systems as consumers of information a new infrastructure is needed. With this in mind, one ought to develop more intelligent business applications tailored to the customer needs. Context-aware information combined with a broader and deeper use of context sources such as sensors, environment

information, indoor sensors and adaptive application behaviour can be factors for creating richer and better applications. Thereby, make the system immediately take better advantage of the information retrieved and pursue the ideal of providing tailored and timely information to mobile information users everywhere. Context-awareness as a valued source of information is also recognized by the Gartner Group (2011c) which predicts that by 2015, companies will generate 50% of their web sales through social presence and mobile applications. They further point at the development of context-aware applications together with mobile-based capabilities as a major step towards this goal.

1.4 Conclusion

Research is about being in front, investigating problems and issues of tomorrow. As highlighted in this chapter, this is an issue this work is concerned with and contributes towards the goals of mobile devices, ubiquitous computing and context-aware applications.

In this research I seek to build on the last two decades of achievements by even further utilize context as a source of information for user information- and interface tailoring. Context-aware solutions and applications have been around for some years, successfully enriching applications (Dey and Abowd, 1999), but earlier approaches have been limited. Often they have only looked at one or a few sources of context-aware information, utilized separately. As a full blown, integrated, context-aware experience, still is an unachieved scenario by mobile technology, this research will argue that context awareness is a key concept to understand how mobile phones can become true companions.

By investigating an in depth implementation of context-awareness it is my goal to research how multi-dimensional context-awareness can be used as information for context reasoning in mobile applications. This would contribute to the knowledge of the field by further extending the utilization of context-awareness. To better cope with resource constrained mobile devices, cloud-computing services could be pursued for offloading work intensive tasks (Simoens et al., 2011). The utilization of cloud services for context-aware information and thereafter incorporating this data in

mobile device has not yet been thoroughly researched and as such this thesis would add to the knowledge base in this area.

Building on Design Research (Hevner et al., 2004) my work proposes an approach, where context-aware information from several sources is combined to build a rich computational foundation. Furthermore, service distribution through the use of cloud-computing technology is researched in pursuit of Weiser's visions to create better ubiquitous computing experience and to seamlessly weave the user experience and invocation of control of the smartphone into peoples' lives.

The structure of this thesis is as follows: Chapter two presents the literature review from the areas of ubiquitous computing, context-aware computing and cloud computing where relevant work are identified and discussed, before the chapter ends by identifying the research aim and objectives.

Chapter three presents and discusses the methodology used. The chosen methodology, Design Research, is detailed and justified. The chapter then explains how the methodology is applied to the overall PhD work as well as for the individual research experiments presented in chapter four, five and six.

In chapter four, the first research experiment of my work is detailed. In this experiment the introduction of a multi-dimensional context-awareness computation is implemented in a personal information manager device. Through practical user evaluation the impact and feasibility of the solution is evaluated.

In chapter five the second research experiment is presented. This experiment is situated in a meeting environment and the development of an intelligent meeting assistant for providing supplemental presentation information is presented and examined. The architecture is discussed and the combination of context-aware information with a scalable cloud environment is investigated through end-user evaluations.

In chapter six the third and last experiment is presented and discussed. In this research experiment cloud computing and context-aware integration is taken one step further towards creating the seamless user experiences by developing the context-aware phone. Building on research from chapter five, this experiment takes the users one step further in having the always tailored and customized phone by adapting the user interface, applications and contacts available, based on context-aware information from cloud services and on device information and sensors.

Finally, chapter seven concludes the research and highlights the research findings and contributions.

Chapter 2

Literature Review

As indicated in chapter one, the main theme of this research is ubiquitous computing and context-awareness in mobile devices. Accordingly, in this chapter the literature review of the research area is identified, presented and discussed, and the chapter concludes with the research aim and objectives.

The chapter starts off from the world of ubiquitous and mobile computing as introduced in chapter one. A more detailed focus will now be applied to the descriptions and the attention is narrowed to focus on mobile systems and their interactions. Thereafter, I will elaborate and justify the use and concepts of cloud computing before returning to mobile computing and looking at how mobile services can be enriched by cloud computing, thereafter, I will investigate the current link between mobile context-awareness and todays cloud services. Before concluding, the chapter ends by highlighting the research gap and presentation of the research question, with its corresponding aim and objectives.

2.1 Mobile computing

This section discusses the key concepts and topics from mobile computing. Mobile devices of today are fast, powerful and feature rich devices. Mobile computing itself is weaved into our everyday lives and equips us with several tools for performing activities on the move. We embrace these capabilities and various innovative uses have been found utilizing this in both social, work or cognitive activities (Tamminen et al., 2004). They all support different applications environments, and the smartphone market for consumers and business have blossomed (Koller et al., 2008). Indeed, overall smartphone sales are increasing enormously, growing 62.7% from 2010 to 2011 and reaching a total of just below 500 million devices in 2011 (Canalys, 2012). In particular the Google Play market for Android devices and the

App Store for iOS devices have increased enormously (DiMarzio, 2008), and are estimated to be worth $17.5 billion by 2012 (Shiels, 2010). They are no longer alone, manufacturers such as Microsoft / Nokia with their market place for Windows Phone and Samsung with their equivalent online shop for their Bada operating system are two out of several manufacturers with, currently, a smaller foothold.

Mobile phones have taken a huge step from their monochrome screens, to today's sophisticated hardware and software. This evolution also increases the complexity of development for these platforms in regards of heterogeneity and language support. With new programming languages emerging, such as Objective-C, C# Windows Presentation Foundation, MeeGo (QT) or HTML5, the corresponding development tools and language ecosystems are also constantly renewed. In this context, Dubochet (2009) researched modern programming languages to see how they affected code style and developer productivity. Many new features available not only affect developers but also the business side and such new features make way for innovation opportunities. Today, touch screen smartphones are commonplace and aid for user groups, such as those in need of accessibility functions, are ever improving. Still, there are challenges left to be addressed. The main concerns and challenges in mobile computing have been constant since Satyanarayanan described them nearly two decades ago (Satyanarayanan, 1996). We summarize the challenges below:

Table 2-1: Challenges in mobile computing (Satyanarayanan, 1996)

Challenges	Descriptions
Mobile elements are resource poor	Although continuously increasing and improving, a mobile unit are inferior to a static unit.
Mobility is inherently hazardous	Being on the move they are more exposed for loss and damage
Connectivity varies in performance and reliability	Connectivity varies with surrounding. I.e. in the city, in the mountains or in a subway.
Mobile elements rely on a finite energy source	Battery life is a constant challenge and there is always a trade-off between power, functionality and battery consumption.

The challenges identified are still valuable points to take into consideration when developing or designing for a mobile environment. For *poor resources*, it is emphasised that compared to a static unit, the mobile will be weaker in terms of computing power, memory, storage or speed. This is as valid today. Even though the units are improving fast they are still inferior to a desktop computer.
Some of these challenges are solved by employing cloud technologies in interplay with mobile devices, but *mobility is hazardous* cannot be neglected. Although industry standards are developed for smartphone devices used in extreme environments, this is not the case for most devices. They are vulnerable to damage, loss and theft.

Mobile connectivity is highly variable was another point in Satyanarayanan's (1996) list. It is still a fact that devices at times are prone to experience connectivity issues. But the rule rather that the exception is that we are used to being connected and live in a world of interconnected, online services. Mobile connectivity is a relevant issue, but it is probably the one bullet point of the list that is closest to being obsolete today. Last but not least, Satyanarayanan pointed the attention at mobile elements relying on a limited energy source. After a period of years where device capabilities grew together with battery life, energy sources in these devices are today a very limited factor and Satyanarayanan's point is more valid now than ever before.

2.1.1 Device opportunities and devices

There have been great evolutions of mobile devices over recent years. Indeed, going back only five years ago makes the current devices look almost futuristic. The top-notch devices in the smartphone segment five years ago were, for a while, the Sony Ericsson P1 and the HTC TYTN series. That was correct up until Apple changed everything with their disclosure of the Apple iPhone on January 9th 2007, with the device hitting the US market in June same year. Up until the launch of the iPhone, smartphones targeted solely the business user segment, and only individuals with an interest high above average for technology would invest in a smartphone at this time. The current leading smartphones focused on business utility applications such as e-mail and office applications (Word processing and spread sheets). The major game changer from the launch of Apple iPhone was the focus on private consumers. This

intensified a year later, when Apple launched the Apple App Store as well as the Apple iPhone 3G in 22 countries worldwide in the summer of 2008.

Following the launch of the Apple iPhone, another competitor entered the field the autumn of 2008, the Android operating system by Google. The launch in September 2008 was the first commercial version of Android. Behind this operating system was the consortium Open Handset Alliance (Open handset alliance, 2007). Together with the launch came the first Android device as well, the HTC dream (HTC, 2012).

Figure 2-1 Left: The first Android phone, HTC Dream. Right: First iPhone, iPhone 2G (GSMArena, 2012)

To give a clearer idea of the evolution of mobile devices from 2005 to 2012, a comparison of state-of-the-art phones from 2005, 2008 and 2012 are offered below (Table 2-2).

Table 2-2: State-of-the-art device comparison

	2005	**2008**		**2012**	
	HP Ipaq 6500	iPhone 3G	HTC Dream	iPhone 4S	HTC One X
RAM	64 MB	128 MB	192 MB	512 MB	1 GB
CPU	312 MHz	412 MHz	528 MHz	Dual core, 1 GHz	Quad core, 1.5 GHz

Common for all of these platforms are their great technological capabilities. These smartphones pack more power and features than what a desktop computer five years ago was capable of. As they are getting more and more features, functions and hardware technologies added on, they become more and more resourceful. For the consumer this only plays a part if the developers, industry and researchers manage to exploit these functions in a meaningful manner. A smart device incorporating vast amounts of information does not necessarily mean that it is intelligent in front of the user. To be able to exploit data from environment, functions, location, sensors, data stores the applications need to be aware of the users' needs. The applications need to be smart and behave in a way that can improve the users' experience or the users' adaptation of the technology or application. The applications need to inhabit context-aware sensibility.

2.1.2 Mobile devices: Personal Information Managers

As previously described, mobile devices have been used in the making of many context-aware systems. This is in spite of the fact that such devices may limit the functionality of the developed application due to limited computational resources on mobile devices, such as battery power, information storage, computing power, and communication.

The world of end users, on the other hand, has changed dramatically. With portable computer technology in combination with wireless communication, the world of end users has been expanded allowing users to bring work and leisure into the world of mobility (Chen and Kotz, 2000). The constant increase in bandwidth enables constant access to personal information, business contacts or public resources. These devices separate from standard mobile phones by some parameters and such devices are today often referred to as Personal Information Managers (PIMs) or smartphones in the world of a ubiquitous computing. A state-of-the-art PIM contains today a rich variety of technologies: a multifaceted operating system, mobile phone, calendar, music/video player, radio, read/write office documents, positioning systems, sensors and a number of wireless communication means. Work conducted by several researchers on mobile devices (Bilandzic et al., 2008) and also some work relating to PIM devices, Perttunen et al. (2005) from the Finnish industry and Spriestersbach et

al. (2001) from SAP Labs, are examples of this. Far less research has been done towards integration of functionality from calendars on such devices. Research done toward PIM units can be exemplified through the work of Cutrell et al. (2006). Another application utilizing context on a mobile phone, though not a PIM device, is exemplified by Schmidt et al. (2001), who produced an application where users, through a dynamic WAP interface, could see the current state of their contacts before calling them. This meant all participants had to share their current status, but would benefit by not having anyone calling them if they were unavailable or busy. This context-sensitive phone book was found useful by a large majority of their test group. One issue with this application is that although it is implemented on a mobile phone, it still needs a server application to control and manage the context and users' status. The mobile device merely did a communication call to retrieve or update the status and was by itself not able to take advantage of its own context. The possibility of further extending the device and creating a valuable user experience could be investigated from the usability angle, but that is a separate field of research and out of the scope of this research.

A rather different approach to a context-aware mobile phone application is the experiment conducted by Häkkilä and Mäntyjärvi (2004) from the Finnish telecom industry. They created a location sensitive system that delivered multimedia messages to user's mobile phone every time s/he approached special locations on a determined route. With location as context-aware input the deliverable to the user consisted of text and / or image with information relevant to the current user context. This approach was dependent on a server to receive the messages. One could imagine a further development of this approach by moving more of the logic to the clients and use a richer mobile client, like a PIM, thus achieving independence of server communication. Server communication could then be limited to receiving updates to locally stored information and perhaps collecting user comments, recommendations etc.

Today every smartphone is also treated as a personal information manager. This leads to new possibilities since we tend to keep it close at hand at all times, as well as keeping a vast amount of personal information on the device. Several attempts of

exploiting such information have been conducted, and a popular example is the use of on device contact information. These are either used as a basis for extracting context-aware information or as a source being manipulated in a context-aware environment, making them more adapted and tailored (Ankolekar et al., 2009; Oulasvirta et al., 2005; Schmidt et al., 2001).

As described, context-awareness has evolved as devices are getting increased computation power and extended hardware specifications. Enriched by sensors, communication capabilities and filled with data, they are prone to be used as sources for planned context-aware exploitation. Accordingly, we will further investigate deeper into context-awareness, looking at definitions and sources of this information.

2.2 Context-aware computing

Advances in computing hardware and software technologies are key factors behind the proliferation of mobile computing and, as described, the paradigm of ubiquitous computing advances the world of mobility and context are combining efforts in what are often described as context-aware computing. Following Weiser's (1993) vision, the most overwhelming technologies are those that eventually make themselves invisible, totally and interchangeably integrated with environment. In his works, Weiser identified two key components for this to succeed and to act as a breeding ground for context-aware services, namely location and communication (Weiser, 1993; Weiser, 1994; Weiser, 1991).

In this respect, one of the earliest context-aware system realizations was ParcTab (Want et al., 1995), which was implemented at the Xerox PARC Research Labs based on the vision of Mark Weiser. One of the main objectives of the project was to design a mobile hardware device, the PARCTAB, which enabled personal communication through architecture with support for mobile communication. The prototype was developed for and tested in an office environment. As part of this prototype application, devices such as a palmtop computer, an electronic notepad and an electronic whiteboard (Figure 2-2) were used.

Figure 2-2: The PARCTAB prototype (Want et al., 1995)

The resulting product of this was a digital office facilitated with some intelligent electronic gadgets. All the devices in use as part of this project were connected to the local area network using infrared (IR) connections, which also enabled the location management of the users. Knowing the location of the user, the relevant communications (e.g. emails and phone calls) were then routed accordingly to the user's physical location or computer terminal.

2.2.1 Definitions of Context

Context and context-aware computing are, since they were introduced to the information systems and computing field, terms with a large diversity of definitions depending on the perspective and circumstances (Loke, 2006). Context-aware computing was discussed early on by Schilit et al. (1994) who looked at the subject from the perspective of a computational model capable of occurring over a large span of locations and situations. They described it as software that "*adapts according to the location of use, the collection of nearby people, hosts, and accessible devices, as well as to changes to such things over time*" (Schilit et al., 1994). Similarly, Chen and Kotz (2000) saw it as a paradigm of mobile computing where applications could discover and take advantage of contextual information such as user location, time of day, nearby people and devices, and user activity.

From a different perspective, Pascoe (1998) defined context as a subset of physical and conceptual states of interest to a particular entity. Rodden et al (1998), focused on that besides the actual location, the environment you are in or are a part of, will

play a role for this particular definition of context. Rodden et al (1998) also agreed that location would be part of context definition, but it should not be limited to that. Schmidt (2000) adhered to this perspective and exemplified several areas of interest for context-aware computing such as environment, situation, state, task and more. He also supported the predominance location has as a source of context-aware information.

Ten years after Weiser's initial discussions of context, Dey and Abowd (Dey, 2001; Dey and Abowd, 1999; Abowd et al., 2002) brought in a more generic description. Thus, Dey (2001) argued that the definitions introduced previously were too specific for the concept of context-aware computing, and proposed new definitions for the terms 'context' and 'context-aware'. Accordingly, he built upon the earlier work and widened the perspective even more by stating that: "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 applications themselves" (Dey, 2001). Based on this definition, a system is context-aware if it uses context to provide relevant information and/or services to the user, where relevancy depends on the tasks in hand. Over time, this has been established as the most used and cited definition of context-aware computing. Nonetheless still today researchers are constantly reformulating the definitions causing diversity in the definition of the field.

Another approach is represented by Dourish (2004) which looked at context from the interaction point of view. In his opinion, the definition of context was by far not as important as the way it was managed and used through user interaction. In Dourish's (2004) opinion, the user's actions were the most precise definition of context.

Wei and Chan (2007) incorporated a decade of work on context-awareness and investigated the matter further. They presented three characteristics of context-aware applications:

- *Context is applications specific*
- *Context is external to applications*
- *Context can be used to change behaviour, data or structures*

In their work, they highlighted how context, as a source of information, represented by these three characteristics, could be used to adapt applications. They explored and categorized the information into physical context, computing contexts and user contexts (Wei and Chan, 2007). The goal of this information is to create a better experience for the user, and the way context-aware information is adopted together with the time of the adoption are major issues to be investigated. The adaptation may happen with data, behaviour or application structures. They argued that the more fundamental the adaptation is, e.g. changing structures, as well as the later the adaptation is, e.g. runtime, the harder it would be to achieve the implementation. The benefits though, might be increased, as the changes are more fundamental.
Satyanarayanan (2011) exemplified context-aware attributes as: physical factors (location, body heat and heart rate), personal records and behavioural patterns. He stated that the real issue was how to exploit this information and how to deal with all the different representations of context.

Over time the research and effort put into defining context seems to have resulted in a broader perspective at every step. This is positive if the goal is generalisation and a general acceptance of the definition, but a broad definition is not necessarily the best solution when it comes to making use of the term context in specific settings in applications. An on-going debate has not necessarily only negative effects. It could be a way to continuously improve the current definition. On the other hand, when building applications supporting standards or when trying to communicate findings to a research field, this vagueness is a limitation and a possible source of misinterpretation. Dey (2001) suggested that a developer aware of the current context would more easily be able to determine the requirements and functionality of the application under development. This issue of ever-increasing number of context dimensions needing new varieties of sensors to be captured is tightly related to developments in software engineering and programming languages and techniques also introduce new challenges for the scenario they realise. Coupling to platforms or implementation languages makes the transmission to other devices/platforms harder and can make the reuse of context extremely difficult (Wei and Chan, 2007). The increasing number of developers and researchers see the context as an important factor when designing new applications. Several research contributions have looked

at the aspect of context implemented in an application. As in the research trying to define context as a term, this research is informed by a surplus of definitions of context and a great variety of implementations.

As described in this section, although different researchers from different periods of time have had several definitions of context, all agree that those working in the field commonly understand the concept of context. Since so many definitions exist, there are as many possibilities for implementation and to make uses of the context. In such a lively environment, there are of course conflicts when defining the user context. Hong et al. (2009) looked into this and identified three different types of conflict to handle: Conflict in sensing context (i.e. GPS location and camera image analyses yield different results), service resource conflict (lack of resources for producing results) and user preferences conflict (if out of two or more users identified to have the same context, only some are produced tailored results due to conflict in preferences). These issues can arise when multiple context-aware sources are combined and are inputs that need further research to establish a pattern for how they should be handled.

Context-aware applications' capability to adjust to an always changing environment is one of the things most in common with ubiquitous computing (Munoz et al., 2003) and, as the world evolves to continuously add new everyday devices with communications capabilities and technology continues to weave into our surroundings (Wei and Chan, 2007), context will continue to play a major part in developed technical solutions. To this aid, context-based services as well are expected to continue manifesting its importance in the upcoming years (Gartner Group, 2011a), making it a relevant factor on clients as well as servers. Nevertheless, and as described so far, most researchers identify context on the basis of one or a few factors, and their importance are unquestionable. In the next section I will explore and elaborate on these most common sources for context-aware information.

2.3 Sources for Context-Aware Information

Having investigated the most common definitions of context-aware computing, I will now proceed by inspecting the most common sources providing context-aware information including location, sensors and activities.

2.3.1 Location as a Source for Context-Awareness

Location has from early on been the primary and often leading source of context-aware information (Wei and Chan, 2007). With the quick spread of global positioning systems (GPS) in all modern smartphones, location is more or less regarded as a standard for determining location (Strobbe et al., 2011). With that said, there are several flaws about GPS positioning as well. Although an excellent technology for outdoor use, it is almost of no use indoors since it relies on line of sight to the satellites, and such an environment requires therefore a different positioning technology. Also in cramped quarters in city centres, GPS positioning can be confusing due to locally lack of signal perception. Due to its imperfectness, GPS is supplemented by other location positioning techniques as well. There are several options for indoors and outdoors positioning, all supplementing GPS position (Kaasinen, 2003). Amongst them, a well-known positioning technique is to use triangulation from the mobile phone network. This identification relies on identification of a mobile phone being with reach of a network cell. The accuracy of this technique will depend on the size of the cell. In urban areas this can be limited to tens or hundreds of meters, whereas in more rural areas the cell size could be much, much larger. One existing technique for addressing this issue is to use information from more than one network cell tower, preferably three or more, to do network cell triangulation. This then can help determining a more exact position by leveraging trigonometry to determine object position.

For indoor positioning there are several approaches available with WIFI technology being by far the most common. This is due to the fact that there is a lot of infrastructures already build for WIFI technology, and the hardware is fairly cheap. Other common indoor positioning techniques include Bluetooth, which has a fairly limited reach, about 10 meters, making them ideal for proximity based scenarios. Lately, both RFID with a reach of some centimetres to a few meters and NFC with a reach of a few centimetres to an inch have become options for detecting clients. However, a drawback of both RFID and NFC is the cost of the equipment (Strobbe et al., 2011).

One early approach concerned with the use of location as source for context-aware information were the works of Abowd et al. in the Cyberguide (Abowd et al., 1997) projects. This project implemented a hand held electronic tourist guide system that supplied the user with context sensitive information. Cyberguide was developed for indoor use at Georgia Tech (Abowd et al., 1997). Through this early work, location was the primary factor for identification in an indoor and outdoor realization of the framework. Long et al. (1996) described research implementations of the Cyberguide framework and reflected on its implementation, use and benefits. The system consisted of several modules, and among them was: information module, location module and communications module. The essential parts of the applications written to suit as best as possible the different experiments were implemented. Issues they encountered and discussed further were platform portability, communication complications (in terms of cross device messaging) and external resource consumption (such as web resources). Still they were able to show a feasibility of using location as a driving factor for context-aware information.

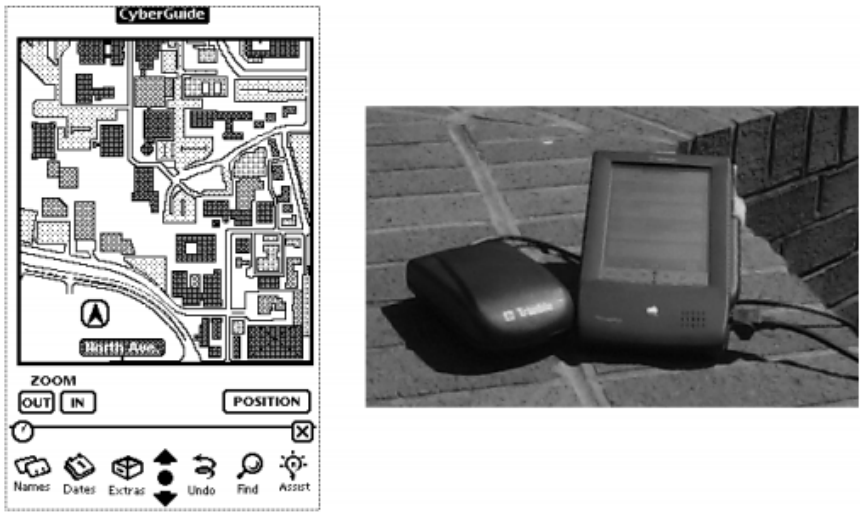


Figure 2-3: Cyberguide implementation (left) and GPS unit (right) (Abowd et al., 1997)

Their efforts were taken further by others who have attempted to use location as a prime factor for context definition such as Davis et al. (2001). They created a mobile context sensitive tour guide which presented information based on personal interests and location. They found the tour guide behaving as intended but the users' reactions to the prototype was negative because they needed to enter too much information before the device could be used.

Location also played a central part in the work of Simcock et al. (2003). In their research, they explored a location based tourist guide application implemented on a Windows CE PDA with off-the-shelf GPS equipment to pinpoint the users' location. With this as a basis, they provided the user with an interface well suited for a handheld device and to be able to produce information closely relevant to the user, i.e., displaying nearby sights. They recognized the use of positioning technology as highly relevant and useful for tourist applications, and presented novel results in terms of user satisfaction for sighting information presented. They admitted that the lack of resources in the handheld device was keeping them from providing a richer multimedia content to the users.

One example use of methods other than GPS for positioning was in the work of Kjeldskov and Paay (2006). In their experiment they created a client-server application where the client was a PDA. Then, while users were moving around in a city they were identified and localized by Bluetooth beacons and information about nearby sights and landmarks was pushed from the server to the device. Although user action defined context, by moving around and performing activities, it was the location information defined by Bluetooth proximity that defined what actions to be taken (what information to be pushed), by the application in their experiment.

From the area of tourism, Echtibi et al. (2009) built a location based context-aware tour-guide, *Murshid*. This application was implemented on mobile devices using J2ME technologies. Their application could take into account several context-aware dimensions, such as location, user interest (stored in profile), time and user interaction. This lead to a diverse foundation to make context-based decisions on, but the driving source was the location. All other context-aware sources were based on location as being part of the information produced. For instance, if a user at a given time updated his interests, then the location was added as a second identifier when the server side was updated with information. Based on this, tailored sightseeing events and places of interest could be highlighted. Location is always identified by GPS, so when a user is out of reach of satellites, the system will have issues with producing up to date information.

Kaasinen (2003) has in her study investigated location as the primary factor for context-aware information and offered thoughts on its suitability. She emphasized the need for comprehensive services, geographical coverage and enough information being available to create realistic and viable resources. She also acknowledged the effort of participating in building rich information sets, by allowing for user interaction with data back to the servers. Full exploitation is first achieved when service density is high and the process of integrating this is made smooth.

2.3.2 Activities as a Source for Context-Awareness

Following definitions from Dourish (2004), one example of user action defining context in an application is the approach taken by Kjeldskov and Paay (2006) in their experiment. They created a client-server application where the client was a PDA. Then, while users moved around in a city they were identified and localized by Bluetooth beacons. While this application and approach was successful in exploiting the users' current location and furthermore allowed the use of information retrieved, I find the application somewhat limited. The definition of context is quite narrow, as only location is taken into account, and the information available is pushed to the client. There are a number of other contextual factors that might be utilised in order to make the application more context-aware rather than just context determined.

Research by Tamminen et al. (2004) had a user-centric approach to a task driven, context-awareness implementation. They based their research on the assumption of context always being determined by the specific situation it is used in, in combination with the activities undertaken by the users. They derived the context from a specific situation by examining how its used, loaded with different action resources, such as motives, plans, people, device and so forth. This very interesting and novel approach places the user at the centre of attention and exploits activities and tasks in a much more user centric way. The findings of Tamminen et al. (2004) highlighted the importance of technology's need to fit with social patterns and activities, and not the other way around.

Following the perspective and ideas from location-aware computing, location-based services can only provide information relevant for the current context if they are

made preference-aware argued Mokbel and Levandoski (2009). In their work they presented the combination of location and activity, helping users get a more personalized response to their queries. Examples of such tailored response can be the search for a restaurant. Your personal preferences will likely influence the search results you wish to be presented (i.e. being a vegetarian you would like to rule out steak houses). Their goal was to make systems behave less rigid and deliver more personalized results and data. Their ideas have general applicability and the work of increasing focus on personalized user profiles, context-aware queries, and their efficient and scalable execution.

Activity centred context-awareness can be woven around a users current tasks. Woerndl et al. (2009) extended this idea into a personal recommender system based on location, user preferences and information available. They argued for incorporating context-awareness into the data model, as the quality and validity of the recommendation will be dependent on the context (Woerndl et al., 2009). They displayed this system in action in a gas station recommender system. Context was extracted from people's actions, i.e. driving directions, speed and from user recommendations (i.e., gas station experience) and from car activities, such as location and gas level. The system was decentralized, letting decisions be ruled on the users' devices and data stored locally to protect privacy. From the implementation of the system and the performed evaluation, they argued for the feasibility of such a system and how this might be applicable to other activities such as restaurant visits, shopping, leisure activities and business. An interesting option would be to see their approach in combination with cloud based computing resources to enhance the recommendation algorithms, whilst still keeping user privacy intact.

2.3.3 Sensors as a Source for Context-Awareness

Sensors have been a potential source of context-aware information from the early days of mobile computing. Early on, mobile (phone) devices were described as ultra-mobile devices (Schmidt et al., 1999b) which mostly referred to PDAs. In this early research on context-aware computing in a mobile scenario, sensors were not yet an integrated part of the devices. Still, they were perceived capable of adding new sources of context-aware information besides location. Schmidt et al (1999b) described this opportunity as sensor fusion and demonstrated how this can be

modelled and exploited in a prototype application. Experiments conducted early on (Schmidt et al., 1999a) exemplified how context information may be changed accordingly to behaviour. This was ultimately implemented in a prototype application for a PDA where backlight and font size are changed according to the active users' context.

Another early approach of incorporating sensors led to the context aware phone (Siewiorek et al., 2003), which produced a prototype that adapted to the environment through sensor information. The accelerometer, light intensity sensor and microphone sensor were included as sources of information. The sensors were mounted on a person's body, and interacted through a small, portable computer module with a mobile phone. Their effort highlighted an early manifestation of sensors as input for context-aware solutions.

Sensors are nowadays increasingly common, also as standard equipment in the state-of-the-art mobile phones. Wei and Chan (2007) refered to them as input sources for defining the physical context. Traditionally, sensors external from the (mobile) device are also covered by their definition, and phenomena to be captured typically include light, noise, motion, acceleration, temperature and similar measurable data. Further sensors, and thereby data sources, are continuously added as manufacturers continue their development. One very interesting aspect of this is its potential for becoming ever more important as more of these sophisticated measurement instruments are incorporated into mobile devices as part of a standard package.

One major issue in the combination of sensors and context-aware computing is the use of device resources. Constant monitoring of sensors may lead to fast and frequent changes in applications status. This leads to a large consumption of resources, which is a challenge on mobile devices with an already limited supply of battery and processing power. Kang et al. (2008) explored this issue in their implementation of a more energy efficient framework for monitoring sensors in a mobile environment in their framework SeeMon. Their ideas incorporated monitoring of context-aware status, but only actions to be taken when the context-aware status was changing, not on updates or events happening within the same context-aware status. This makes

SeeMon a framework for scalable and energy efficient management of context-aware statuses. So far, their implementation is exemplified on an ultra-portable computer and a custom tailored and designed wearable mobile device. An interesting expansion of their work would be to evaluate its performance with a state-of-the-art mobile smartphone device, which now incorporates a wide variety of sensors and is capable of detecting context-aware statuses given the right software applications.
There have been several research contributions studying the technology behind these sensors, but not as much on the actual use of these (Lovett and O'Neill, 2010). For instance, Parviainen et al. (2006) also used sensor in their works namely in a meeting room scenario. They found several uses for a sound localization system, such as: automatic translation to another language, retrieval of specific topics, and summarization of meetings in a human-readable form. In their work, they found sensors a viable source of information, but also acknowledged there is still work to do, like improving integration.

Modern smartphones have a number of built in sensors. For example, the HTC Nexus One has five sensors available: accelerometer, magnetic field, orientation, proximity and light. All of these sensors can be accessed as local services through well-defined interfaces in the Android platform. By combining the aspect of sensors with context-awareness and integrating it in a mobile application, one reduces the workload for the end users. Thus, in combination with a cloud-based server application one is able to remotely configure the mobile devices independent of the device types. This creates a new concept of context-awareness and embraces the user in ways previously unavailable.

2.3.4 User Profiling and Personalization as a Source for Context-Awareness

User profiling can generate valuable information to be used as the basis for making context-aware decisions. One issue is though that in order to create an accurate user profile it takes a lot of effort. This is often due to the fact that users are reluctant to give away their personal information and are not eager to fill in forms or give explicit feedback (Strobbe et al., 2011). The ideal way to create the user profile is thus by automatically and invisibly collecting the information (Strobbe et al., 2011). Examples of such a user profiling system are the Amazon and EBay recommendation systems.

Another approach looking at the cognitive elements of context is described by Prekop and Burnett (2003). They argued for the development of context-aware applications supporting cognitive activities. They proposed an activity-centred context process and described this at a conceptual level. What is very interesting about their approach is the activity-focused approach that also acknowledged that an activity could be surrounded by context from the environment. Although they did not propose how to solve this technically they did agree to difficulties in defining and extracting environment information.

Another approach on user profiles, looked upon from a social science inspired perspective, is the work from Dourish (2004) and Sarker and Wells (2003). From their classifications and descriptions, it is stated that the user activities are important. Thereby, having a series of user interactions can lead to observations and recording of behaviour that again can be used for profiling the user or the context-aware dimensions s/he is interacting with. Bush et al (2006) took this one step further by implementing autonomous agents operating on behalf of the user. Users are not only profiled and have tailored information presented, but rather a system of agents and nodes constantly monitor the user environment to be aware of content types and device capabilities. Through a *Personal Assistant Agent* digital content and applications were distributed in a heterogeneous wireless environment.

The significance of exploiting human emotions and presence together with location is highlighted as a possible way forward by Wei and Chan (2007). This can be used to create a richer basis for context-aware adaptation and be a viable exploitation of a new source of context-awareness. Still, one needs to carefully think through several factors about adaptation such as what to adapt, how to adapt and, last but not least, when to adapt (Wei and Chan, 2007).

The use of profiling and personal adaptation is interesting and presents many opportunities for novel applications. Even though it is a very popular dimension of context-aware information to exploit, it is not always without uncertainty. Users of context-aware applications can hail from a variety of backgrounds and have very

different opinions on how context-aware applications should act and Henricksen and Indulska (2005) claimed that personalization of context-aware applications is much harder than personalization of traditional applications due to the diversity in perception from different users. They categorized personalisation into three categories:

- *End-user programming approach*: The application itself is not pre-constructed but rather assembled and programmed by the end-users themselves
- *User modelling/machine learning approach*: Algorithms based on user interaction history and historical data creates a foundation upon which application content is tailored or changed
- *Preference-based approach*: Based on user driven configuration or interface manipulation of settings, the application is adapted and tailored.

Henricksen and Indulska's (2005) work focused on extending the preference based approach and took into account the possible individual interpretation of context-aware information. To ensure a best possible adaptation of context and thereby the best suitable application for the end user, they offered some general guidelines:

- *Constrain the types of personalisation possible*
- *Integrate personalisation into everyday use of the application*
- *Provide feedback*
- *Make default behaviour useful*

As they highlighted through their work, it is the user satisfaction one seeks to satisfy. It is therefore important that mechanisms for personalisation and tailoring are put into place to create a better experience (use of the application) for the user. Adaptation and personalization parameters are determined, at the time of the service request, by policy rules defined by both the user and the service provider, and managed by their respective profile managers.

2.3.5 Architectures and Toolkits for Context-Aware Information

As described, there are several approaches for integrating context-aware information, as well as many ways to define this source of information. To help accommodate this, a number of architectures and frameworks or middleware have been developed.

One can ask why this is a desired way of designing an application. Early research by Dey et al. (1999) highlighted arguments for this approach. Firstly, the separation between the context-sensing mechanism (possibly a sensor) and the logic implementing it can cause a looser binding between code and hardware, allowing for easier change of context-aware sensing mechanism. This also supports low coupling and high cohesion at code level, which again is good for portability. Secondly, they argued for the possible integration of multiple input devices, finally, we add their argument for how this leads the architecture to scaffold the context-aware input and create the possibility for adding and removing context-aware sources as one pleases. Today's multi-device, cross platform heterogeneous environments have not made these arguments less valid. Therefore, in this section, I will give an introduction and overview of some of the most important architectures and toolkits / middleware.

As previously mentioned in the location section, the *Cyberguide* (Abowd et al., 1997), is one of the earliest examples of combining indoor and outdoor positioning technologies and an early architectural approach for creating context-aware applications based on location and orientation. This approach created a context-aware user centred tour guide. Hardware set up based on a PDA, GPS and network connectivity enabled detection of context and distribution of web based information. This system was initially implemented as a building guide for the department open days at the Georgia Institute of Technology and then improved to guide its users throughout the whole city of Atlanta. The core functionality of the system is divided into four components: *a map component*: organizing the geographical map data and containing points of interest and pathways; an *information component* providing information about the sights one might encounter during a visit. This component is also responsible for answering specific questions of the user such as "Who works in this office?" or "Who painted this picture?"; thirdly, a *Navigator component* responsible for user location identification; and finally, a *Messenger component* i.e. a central module for communicating with other people in the area or with the guide of a specific tour. Evaluation of the system was performed with users, and highlighted issues with respect to the positioning at indoor locations and on-screen maps representations.

Dey et al. (2001) argued that the main problem with context-aware applications was that there were no architecture blueprints supporting the construction of such applications. One can easily understand this argument and one can only imagine how hard it would be to create unison architecture for such applications. This is because context, thereby context-aware applications, is heavily dependent on users' current location, activity, nearby objects and other elements in the different definitions of context. To enable easier use of context, the work was further researched and the *Context Toolkit* was developed (Dey et al., 2001). The proposed toolkit was based on building blocks where all context information was hidden within different components, thereby hiding the complexity of the retrieval of context through the use of sensors. Though they achieved separation between application logic and context retrieval issues still remained unsolved by this approach. As a realization of the context-aware toolkit, Dey and Abowd (2000) built the CybreMinder solution. In this project, they created a reminder creation and delivery system. A user could add reminders with triggers from dimensions of time or location. A typical reminder could be "Remind me to bring an umbrella if the forecast calls for rain". This would be delivered at a given time in the morning at a location given by the user (i.e. at home) if the precondition in the reminder was met (given it rains). As one of the earliest realisations of applications based on the context-aware toolkit, they offered some conclusions. The system was a successful realisation of the toolkit, with its pros and cons. Future research should involve more devices, different means for input and more flexible context-aware basis.

As a result of many frameworks being developed for desktop environments, or for generic situations, Hofer et al. (2003) saw the need for tailoring a framework more specifically to mobile devices and introduced the Hydrogen architecture. Hofer et al. argued that context-awareness was especially interesting for mobile devices since the environment was ever changing and applications had to deal with the constraints of mobile devices in terms of presentation and interaction abilities and communication restrictions. Based on a three-layered architecture, they isolated responsibility in each of the layers: adaptation, management and application (Figure 2-4).

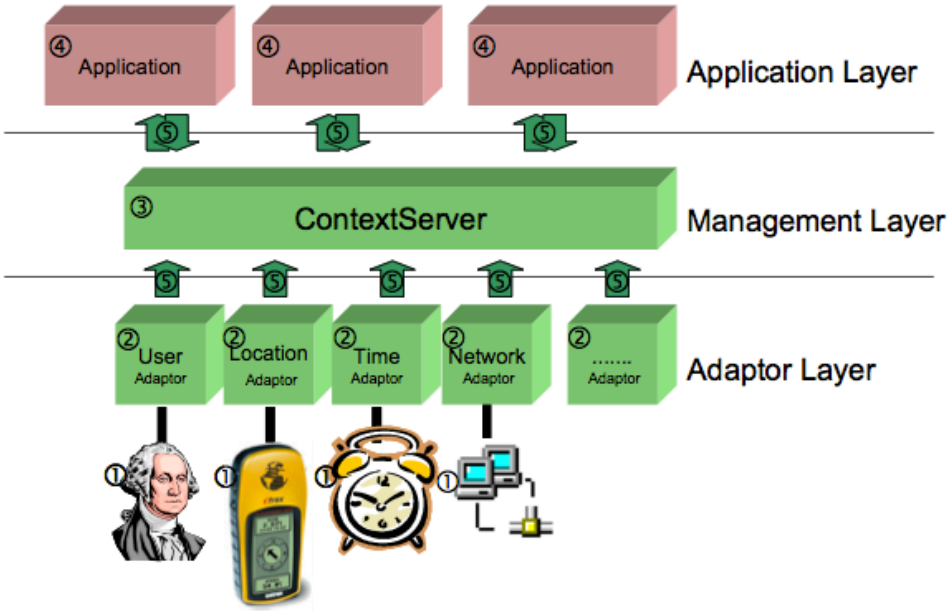


Figure 2-4: Hydrogen architecture (Hofer et al., 2003)

One advantage of their framework is the possibility to have devices speaking to each other in terms of getting context-aware information. By doing so, the context can be obtained without interacting with a server. Architectural-wise, each mobile client will have each of the three layers embedded, making the architecture and thereby the clients use, more robust. Through implementations and testing in a scenario, the framework is evaluated and its feasibility shown. The authors also point at lightweight, extensibility, robustness and capable of handling meta-data as success factors for a mobile context-aware application (framework).

The Context management framework by Korpipaa et al. (2003) takes an ontology based approach to creating their framework. This is quite an interesting approach, and it differentiates from its competitors, since it can apply interpretation on context in real time. An event driven model, is the foundation upon which a traditional client / server architecture is built. The main components of the system consisted of the *context manager* to deal with identification and delivery of context-aware information, the *resource server* for acquiring context-aware data from sensors and interpreting it, the *context recognition service* for conversion between data and ontology and finally the *change detection service* for monitoring service changes which again might imply context changes. Their main components were not particularly different from other approaches, but the implementation of ontologies

gave them interesting opportunities, which they exploited in the data presentation. A main challenge with their architecture is the use of a centralized server. Whilst this solves the needs for computational power in ontology reasoning, it creates a major down side, when (if) the server is down and unable to respond to client interaction.

Work from Biegel and Cahill (2004) described the Cortex framework. Their main goal was to ease the implementation burden for the developers by enabling high-level abstractions and access to sensors. They based the work on a 'sentient' object, which had characteristics such as the ability to sense the environment, operate automatically and react proactively (Biegel and Cahill, 2004). Although their framework is a novel approach, there are questions asked in respect of their extendibility into new areas of context-awareness, as their context-aware information model is somewhat limited and may require further adaptation in each task.

Cass is a middleware to be used for developing context-aware applications (Fahy and Siobhan, 2004). This modular built approach consists of mobile clients with context-aware sensors communicating with a server instance (Figure 2-5). The server has a context-aware interpreter and rule engine implemented, and is able to decide on which appropriate actions to take according to context. Letting the server handle context reduces off the amount of logic needed within client application code. This again decreases the complexity of the client code, making it easier to work with. Moreover, this also supports the separation of concern, as highlighted by Dey et al. (1999).

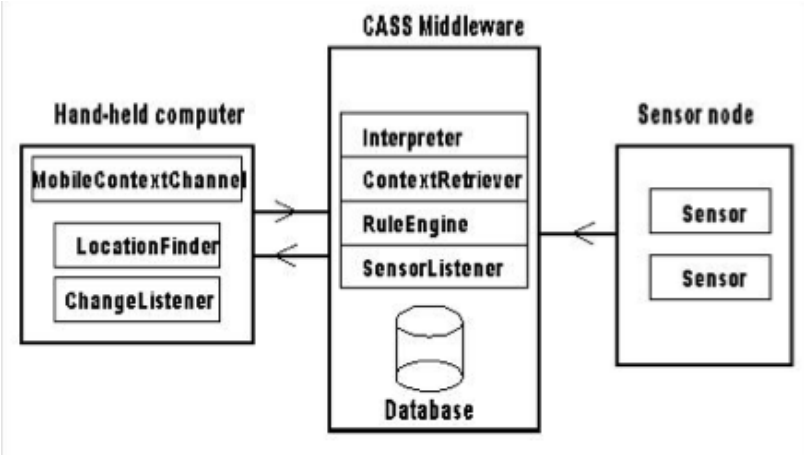


Figure 2-5 Cass system architecture (Fahy and Siobhan, 2004)

The Cass middleware supports context-aware applications on hand-held and other small computers. The server-oriented architecture with loosely coupled clients and sensors supports a large number of context inputs and enables for high-level abstractions while memory- and processor intense work are loaded on the server. This makes the framework a good companion for less powerful and resource constrained mobile devices.

Another work with context-ware mobile applications is the *Nimbus* framework from Muralidharan and Puneet (2006). They have created an interesting task aware service framework. This framework's approach is a bit different in that it is a server-side framework, with small, thin clients. The goal of the framework is to put the user in centre and create the concept of any device, any task, or any time workflow. Through the use of platform managers, context monitor and task handlers, the user can experience how it is to work on a task on one device (i.e. a laptop) for later to continue where s/he left the task on another device (i.e. the mobile phone) on the way home from work. This takes the concept of context-awareness to a level of task-awareness. One can argue that this limits the scope of possible context-aware dimensions, but for sure it creates a transparent and merged concept of tasks in the user world and follows Weiser's (1991) visions for pervasive computing to vanish into the background.

Memory and processing power are highly restricted for applications mobile devices. Enterprises have solved this challenge by moving information-intensive applications to the server side and having mobile clients interact through end points (Shrestha, 2010), still allowing them instant access to relevant applications, services and data anywhere, anytime. For context-aware applications the solution is not as straight forward. Challenges such as monitoring the mobile user context, addressing limited storage and processing capacity and allowing for lightweight deployment of applications need to be handled. Shrestha (2010) proposed a lightweight framework for solving these issues, the *MobileSOA* framework. Based on a context interpreter, service provider and context database man combined context-aware detection and serving services in the mobile world with benefits from Service Oriented Architecture. The framework contained several feasible and novel additions to current context-aware frameworks, but issues such as how to best integrate cloud

services in a context-aware mobile application and how to initiate interaction with new mobile devices were still left open.

2.4 Context-Aware Applications – further Examples and Realizations

So far, different context-aware descriptions and definitions have been described. Alongside this different sources of context-aware information have been detailed and investigated, before frameworks for realization of context-aware applications were discussed. This section will further illustrate the use of context-awareness and its benefits and challenges in different thematic realisations such as tourism, health and in meeting scenarios.

One example of an application taking context into account is the real estate search program (RIA) developed by Zhou et al. (2006) at the IBM research centre. The users' sequential interaction history and his/her interests form the basis upon which the next step of search for information was built. This was implemented and tested as a desktop application. Others have looked at context-aware implementations used in mobile applications. For example Ludford et al. (2006) looked at the use of context to provide useful information to the user on their mobile phone. In their example they produced a reminder / message system delivering messages based on the context of location or (and) time of the day. The application system was standalone and not integrated with an existing calendar system.

Efforts have also been made to make use of context as a tool for supporting business travellers. Hertzog and Torrens (2004) pointed out how applications providing meaningful information based on context would improve the daily life of the business traveller. They produced a program, "Pocket Reality", for a PDA device to help a traveller schedule appointments and avoid conflicting items. In addition, the system provided with environment information i.e. the delay time of an upcoming flight was scheduled in the device. The limitations of this program are that its major feature is calendar scheduling. Hertzog and Torrens' (2004) definition of context, though, is the user's planned activities in combination with the locations, is quite interesting. This is because it generates quite a lot of information about the user, but information is of reduced interest if we have no way to make use of it. Along with

the work of Zhou et al. (2006), these are just two examples out of many and one could only imagine all the other possible scenarios. What we do see is that the business connection towards context is very weak and the definitions of context and context aware tend to be vague. Another approach of merging the worlds of research and business in context-aware computing is provided by Dey and Abowd (1999). They stated that if we fully understood context in a given environment and setting, we would then be better able to choose what context-aware behaviours to support in applications. This could lead to more realistic applications and thereby applications more meaningful to the users. This is also exemplified by Edwards (2005) when he used context information to build an application. In his application, different layers represented different sources of information and they could be reused in later settings. Edwards argued that context is a major part of our daily life and computing with support for sharing and using contextual information (context-aware computing) would improve the user interaction. When viewing people rather than systems as consumers of information a new infrastructure is needed. With this in mind one ought to develop more intelligent business applications tailored to the customer needs.

2.4.1 Context-Awareness in Health Care

The Hospital environment is a reoccurring example of a rapidly changing setting where context-aware computing becomes useful. Hospitals information challenge, both in respect of the number of employees and physicians involved in a treatment but also in regards to the vast amount of information generated each day, are enormous. The advantages of adaptable applications in such an environment is rich if one manages to interpret and manage the massive messaging (Gkonis et al., 2011). Munoz et al (2003) showed an example of how one could ease everyday life in an institution by involving context-aware PDAs which adapted to agents, peripherals and servers.

Another example from the health-sector is a study conducted by Philipose et al. (2004) on activities of daily living for elderly people. They introduced the combination of sensor information with user activities to unobtrusively gather context-aware information. The goal was to be able to monitor people's behaviour to observe if the elderly performed daily activities correct. By tagging everyday objects

with RFID tags, they were able to monitor the use of appliances and objects in a house. As a result of gathering this information they were able to proactively have computer generated probabilistic reports identifying patterns of use as well as user behaviour and possible disorder.

A wider approach to context-awareness is undertaken by Wojciechowski (2009) who investigated this in relation to assisted living for elderly people. By equipping homes with context-aware sensors they managed to integrate services and improved the assistance given. Such an automation infrastructure enabled devices to connect and integrate. A three-layered context model organized the infrastructure. This enabled individual tailoring of the content of each layer, but still hiding the technical aspects and infrastructure from the user. In the *user interface layer*, they enabled custom tailored panels and objects / appliances. User behaviour from interaction in this layer was translated to a corresponding service in the *service adoption layer*. This layer was the second layer of abstraction in the system, still hiding technical details of, e.g., sensors. Query composition and execution on the contained data structures were initiated from here, as well as validation of information and refining of context-aware request from the interface layer. These were received by the infrastructure layer and as described by Wojciechowski (2009), this layer integrated all context-aware sensors. Entity composition, sensor registrations and behaviour selection upon requests were important features. From the viewpoint of an inhabitant of the house, this system facilitated a safer surrounding that assisted in daily activities.

2.4.2 Context-Awareness in Tourism

In tourism we also see footprints from context-aware computing. Many examples use PDAs or PIM's and rely on GPS data for information transfer. This limits the amount of data to be transferred and interpreted on the device if compared to the approach by Kenteris et al (2006) who implemented a mobile tourist guide based on information from specially tailored web pages to create an adapted mobile application (JME) on the fly to be downloaded and installed on their device. Moreover, after installation no network coverage was needed to use the applications. Although a novel approach, it has its limitations. Nowadays with increasing bandwidth available and a rapidly increasing number of Wi-Fi hotspots in reach, we can almost take a high-speed, low cost connection for granted. This would greatly improve the situation as described by

Kenteris et al. (2006) and partly close the gap they described of constant demand of WIFI connection to acquire online content.

Another example is OneBusAway, a suite of transit traveller information tools, which provide, real-time arrival information (Ferris et al., 2010), a trip planner, a schedule and route browser, and a transit-friendly destination finder for Seattle-area bus riders. Ferris et al. (2010) developed a location-aware, native, iPhone application for OneBusAway that leveraged the localization technology in modern mobile devices. Their application communicated with a OneBusAway back-end server over the phone's network connection to request information about stops in a given area, information about particular routes, and, ultimately, real-time arrival information for specific stops. Although the devices get smarter, the technology more sophisticated and the sources of context-aware information expand rapidly it would be wise to keep in mind some simple principles and guidelines as noted by Cheverst et al (2001) in their work with a context-aware tour guide (Cheverst et al., 2000). They highlighted the importance of: simplifying the tasks the user is required to perform in the application, keep application output to a minimum and reducing the complexity of the required mental model of the application.

Schwinger et al. (2005) summarized several of the research efforts in a mobile tour guide paradigm in their survey. By comparing several of the available frameworks / solutions, they inspected what state-of-the-art and identified current issues. From their study they presented several lessons learned and points for further consideration, from which some important are highlighted here:

- The balance between server applications and thin/thick clients are challenging
- More external content (data) can be exploited
- Combining context-aware properties not fully exploited
- Push-based solutions not widely common

They strongly encouraged further research into these areas and potentially develop frameworks for the creation of context-aware applications that incorporate more of these features dynamically. Lastly, mention must be given to the work of Kabassi (2010). His work supported this view and has developed guidelines for creating

personalized recommendation systems for tourists. One major point made supported the conclusions by Schwinger by arguing for a comprehensive use of external data. Additionally, to be successful, these data needed careful tailoring to a user situation in order to be contextually correctly exploited in the application and provide added value for the user.

2.4.3 Context-Awareness in Work Environments

Computer systems can understand the user's actions and intentions in the physical space, and provide appropriate services from the devices to the users automatically. This makes context-aware applications ideally suited for creating smart meeting rooms and active work environments.

Microsoft research on integrating context into applications exploiting desktop calendars can be found in the work of Cutrell et al. (2006). They created an application integrated with the Microsoft Outlook calendar on a desktop computer. Through extending the interface they created a new graphical interface for searching in files and email. What is very interesting in this approach is that they introduce a tag library to tag email and file elements with a context label meaningful to the user. By using labels from a context familiar to the user, they attempted to ease the search and filtering of the tagged objects. A disadvantage in their application is that with such a huge tag library, it takes a lot of time to maintain it. In addition new elements are not discovered in searches before they are tagged properly, which then employs a time penalty on new system objects. However, disregarding this and focusing on the use of the context term, their actual implementation is useful for the users.

There are several contributions that focus on context aware applications in a meeting. Chen et al. (2004) proposed the use of Bluetooth by creating a smart meeting room. The idea was to provide relevant services and information to the meeting participants based on their context. The services included presentation-, lighting control- and greeting service, where Bluetooth was used to register the participants that were in the meeting room. Another research contribution investigating the aspects of localization and identification was that of Bourcier et al. (2007), who focused on the notion of service oriented computing in the meaning of automated discovery and connection to devices. Through the work presented, Bourcier et al. (2007)

highlighted the importance of connecting heterogeneous devices through a generic protocol and strategy.

Smart meeting rooms can also assist with managing the schedule of the meeting attendees, updating related documents during the meeting and providing a communication facility. Accordingly, Ahmed et al. (2005) presented in their work such a *Smart Meeting Room*. This meeting environment was not only able to detect the beginning and end of a meeting, but also provided support of participants' context information, knowledge usability and ephemeral group communication. Their contribution investigated how participant support and user clusters could be facilitated in an intelligent meeting room scenario. The concept of user clusters was interesting and they highlighted how users from different clusters could find ad hoc groups and collaborate. Effortless meeting scheduling, room allocation and updating related documents during the meeting and providing a communication facility were demonstrated. Information distribution was also amongst the goals of their research project and they showed how user groups could receive and collaborate around information within a secure and reliable setting. Context and situation information were aggregated using location techniques. Importance in the solution was focused on providing these services as a separate higher layer, free from low-level implementation challenges. Other low level functions such as automatically muting the phone, authorization checks on documents were also hidden and handled automatically for the user (Ahmed et al., 2005). The proposed solution used the Microsoft .NET compact framework.

In a smart meeting room context, it is especially important that computing devices need to be transparent to the users and the services should be non-intrusive. In this spirit Dai and Xu (2008) presented a Dynamic Context Model to solve the problem of context awareness toward group meeting analysis and services, which included multimodal analysis of group interaction scenarios and provision of attentive services to the users. They aimed to achieve online analysis and services during group meetings. The objective was to make a computing system understand the current context of group meetings based on the audio-visual information at each step and

provide intelligent services instantly to participants. Inside the meeting room, cameras and microphones were placed to collect information during the meeting.
Indeed, in a meeting room scenario, there are a variety of applications in which a sound source localization system may be useful. Automatic translation to another language, retrieval of specific topics and summarization of meetings in a human-readable form are a few of the possibilities. Moreover, one can identify a participant who is speaking and at which time, as well as performing speech activity detection and source localization, all briefly described by Parviainen et al. (2006) in the context of the system developed in their research. The data from their work were comprised of meeting sessions that took place in a room equipped with microphones and other recording hardware. The system was based on spatially separate sensor stations, and each sensor was able to determine the direction of arrival of a sound sources. The results from the test sessions were good, but there were still a few remaining issues like integrating the solution with speech activity detection technique (Parviainen et al., 2006).

There are also attempts to take the smart meeting rooms even further by adding the possibility for a virtual meeting. Nijholt et al. (2006) discussed how to go from captured data in a smart meeting room situation to a virtual meeting room. They looked at issues including turn taking and gaze behaviour of meeting participants, influence and talkativeness, and virtual embodied representations of meeting participants. Their work proposed a visualization of meeting events in virtual reality, interpretations of these meeting events in order to produce semantic preserving transformations and presentation of these meeting events using various media sources. The idea was that virtual meeting participants could mimic what was happening in the physical meeting room (Nijholt et al., 2006). The technology used within the virtual meeting room was considerably different from normal video conferencing by sending the data in a format that enabled analysis and transformation.

Microsoft research on integrating context into applications exploiting desktop calendars can be found in the work of Cutrell et al. (2006). They created an application integrated with the Microsoft Outlook calendar on a desktop computer.

Through extending the interface they created a new graphical interface for searching in files and email. What is very interesting in this approach is that they introduce a tag library used to tag email and file elements with a context label meaningful to the user. By using labels from a context familiar to the user, they attempted to ease the search and filtering of the tagged objects. A disadvantage in their application is that with such a huge tag library, it takes considerable amount of time to maintain it. In addition, new elements were not discovered in searches before they were tagged properly, which then employed a time penalty on new system objects.

In related work, Göker et al. (2004) showed a context-aware information system intended for mobile users. Their research demonstrated a special-purpose hardware device, called '*context tags*', which could work with mobile devices such as mobile phones, to provide ambient information to users on the move. A tag consisted of hardware and software and it could detect the proximity of handheld devices. The result was that relevant content could automatically be distributed and delivered to each mobile phone in the vicinity of physical objects, rooms, and open areas. Context was in general constructed through the user profiles (typically local to a device) and the wireless tags could be embedded in the surroundings. The overall system architecture consisted of three main cornerstones: the content service provider, the mobile user, and the context tag.

2.4.4 Challenges in Context-Awareness

The general approach for context-aware applications is as highlighted, to focus predominantly on one dimension of information. This can result in applications behaving adaptively, but not always making the correct decisions, as there are parts of the picture left out. Frameworks try to compensate for this by elevating several sources of information. Still the effort has its limitations when it comes to generalizability, cross platform communication and adaptations. Brown and Randell (2004) looked into challenges in exploiting context-aware information and applied a critical point of view. Through exploring three scenarios they drew out some limitations in the technology and emphasized what could be done to improve the situation.

- *Provide simple structures the user can interact with*: By providing simple structures, the user can interact with the technology and they can be integrated more closely to daily user technologies. This facilitates easier feedback, and more detailed control of context.
- *Use context defensively:* Rather than actively trying to understand and interpret the user context, applications should take care of using context defensively. Incorrect behaviour should not lead to accidents, malfunction or annoyance.
- *Communicate context to users*: By communicating context to the user this can help shape understanding and use of the system. Users thus become more aware of how to express their actions in a way which promotes context-aware understanding

Also, as summarized by Lovett and O´Neill (2010), context-awareness has become a particular interesting feature of modern smartphones. The innovation in faster CPU and GPU allows for running richer and more intuitive user interface applications. They further pointed at location as the primary source of context, but emphasised the importance of combining it with other sources such as sensors. The following list summarises challenges from context-awareness (Lovett and O'Neill, 2010):

- Mobile applications must use context-awareness from several sources
- Physical, virtual and logical sensors should be used for acquiring context data
- Mechanisms to deal with sensing and inference on device / in services

Also highlighting challenges for context-aware and ubiquitous computing is Schmidt (2010). He gave an overview of how well integrated technologies today are. Still he argued there are issues left to address before we reach a fully seamless integrated technology scenarios. Issues such as sensing, sharing in communities and efforts for avoiding mal-functions are all, important factors to continue to investigate how we cope with technology, which is inevitably an unalterable part of our everyday lives.

2.5 Cloud computing

Through the review of context-aware computing, one highlighted issue is the use of external services as part of the computational foundation. One mechanism for dealing with such a distributed environment is cloud computing. Cloud computing is an emerging paradigm (Mell and Grance, 2011), and has only recent years been gaining a foothold (Mei et al., 2008). Cloud computing is closely related to areas such as: distributed computing, grid computing, software as a service, and cluster computing. All are examples of diversity of expressions for cloud computing and its related areas, describing both the services over internet and the hardware/software systems in data centres providing these services (Armbrust et al., 2010). Regardless of what one calls it, the phenomena are about providing scalable, distributed computer services as needed (Leiba, 2009). The aim of cloud computing is to present a service layer for its users where all detailed logic is made transparent and drawn upon as needed. In general cloud computing is recognized as an infrastructure where all underlying resources (storage, RAM, processors, load balancers etc.) are completely abstracted from the end user. This leads to the cloud provider/vendor to be in charge of performance, reliability and scalability.

Leiba (2009) asked an interesting question regarding whether or not we are back to the early days of computing using thin clients communicating with the backend. He has valid points, but a big difference is in the diversity of today's clients and in the amount of federated service composition in the backend. Borenstein and Blake (2011) also supported the arguments that cloud computing does not consist of any fundamentally new technology as such. They argued as Leiba (2009) that this is a new name for an old trend – centralization of IT operations. However, they pointed at issues concerning standards as vital work for this to be established as a success (Borenstein and Blake, 2011). Still there are major differences between the concept of cloud computing and distributed computing. Even though cloud computing will build on principles from distributed computing the abstraction of resources, as described above, is a major difference. Distributed computing can be characterized as the task of dividing a problem into smaller pieces and have two or more networked computers process the different pieces. Distributed computing can then be summarized as computing orchestrated between two or more computers, whereas

cloud computing is a specialized form of distributed computing which often would be resolved through Internet based services.

National Institute of Standards and Technology (NIST, USA) have proposed a definition of cloud computing, built on five characteristics (Mell and Grance, 2011) (Figure 2-6).

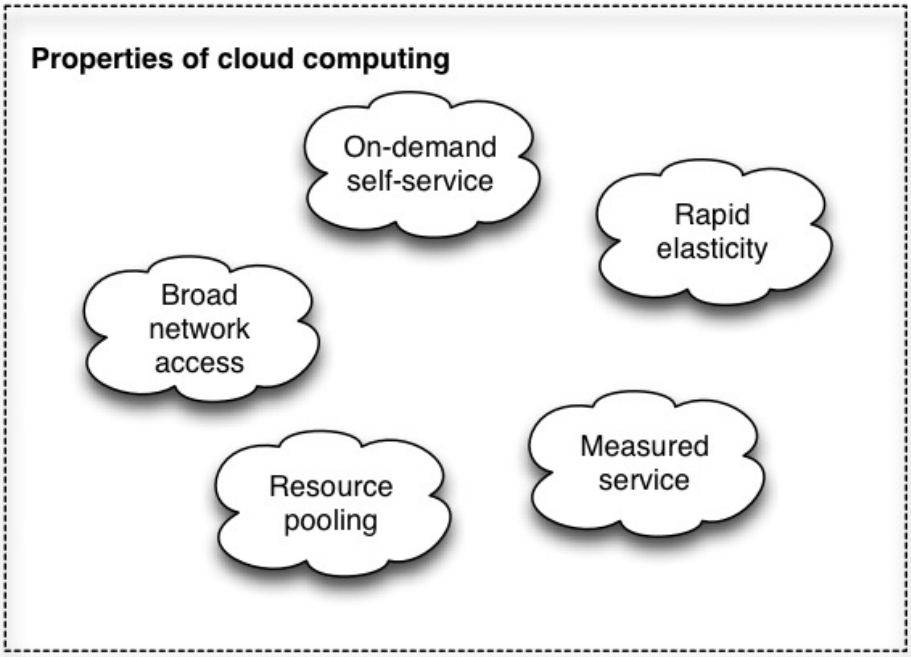


Figure 2-6: Properties of cloud computing

Their characteristic of *on-demand self-service* includes customization of hardware properties without the need for human interaction with the service provider. Broad network access includes facilitation from the provider for using standard protocols for client interaction. This will lead to enabling the use of servers, laptops, smartphones etc., for communication with the cloud. Resource pooling is defined as the ability to serve a large amount of customers simultaneously and automatically redirect or redistribute client accesses based on load combined with location. All these mechanisms should be hidden from the clients' perspective. Rapid elasticity deals with the speed of which services are scaled up and down according to need and load. It is important for this to be consistent and deliver constant quality and to be able to control this metrics should be applied for measuring service reliability and performance (Mell and Grance, 2011).

Cloud computing is characterised by a large number of nodes connected in a network. These nodes can consist of servers, laptops, web services or computer centres (Mei et al., 2008). This network of nodes is called a cloud and applications utilizing this cloud are defined as cloud applications. Cloud is known for having vast amount of resources in terms of computing power, memory and storage. Additionally, as it is an interconnected web of nodes, the scalability is high. This makes it possible to distribute storage and computations over a scalable network of nodes. Large IT companies like Microsoft (The Azure cloud), Google (App Engine cloud) and IBM, all have initiatives relating to cloud computing (Mei et al., 2008). Other companies, such as Amazon and Heroku, offer out of the box public cloud environments to end users for easy access to infinite computing power and scalability on demand (Armbrust et al., 2010). Pricing models on these services are often flexible, and in a common set-up the user pays for the amount of computing power, storage space and up time needed. The calculation for user needs in such an environment can be detailed by the user but is often left transparent to the user achieving scalability. One overall goal of cloud computing is to provide an abstracted layer for application deployment with seemingly endless scalability and more resources for less money, than that available in in-house computer centres (Binnig et al., 2009). Many companies of today look to cloud for a more reliable, cheaper and easier configurable solution than running their own data centres.

Mei et al. (2008) have investigated future research topics in the area of cloud computing. In their work they looked closely at the connection between cloud-computing, service computing and ubiquitous (pervasive) computing. A brief summary is offered in Table 2-3.

Table 2-3: Cloud-, service- and ubiquitous computing characteristics (Mei et al., 2008)

Domain	Characteristics
Cloud computing	- A user sends request and the cloud answers - Calculations are performed in the cloud or in connected clouds - The cloud handles data storage
Service computing	- A service requests and a service responds - Calculations are performed within each service - Data are stored at specific service hosts
Ubiquitous (pervasive) computing	- Request/response are interaction based - Data are (often) stored locally with the application - Calculations are mainly performed by the embedded entity

Accordingly, Mei et al. (2008) highlighted four central research topics for further research: pluggable computing entities to cloud applications, data access transparency, adaptive behaviour of cloud applications and automatic discovery of application quality. This is line with the thoughts of Armbrust et al. (2010) who explored several issues around cloud computing and investigated how one should utilize cloud computing as well as how cloud economics could be incorporated. Armbrust et al. also defined the public cloud versus the private cloud. Public cloud was defined as cloud services made available for end users in an on demand fashion. The user pays as he goes according to the amount of storage space or computing power needed. The Amazon EC 2 cloud (Amazon, 2012) or Google App Engine (Google, n.d.-b) are examples of such. The private cloud on the other hand, was defined as being the internal data centres of businesses or organizations. These are not publicly available and are used internally as computer clouds. In their work they conceptualized the cloud to user relationship as software as a service (SaaS) delivered over the Internet. In such a scenario, a first instance is utilizing the cloud as a cloud user. This user can then make new applications available through SaaS to consumers (SaaS Users). An exemplification is given below (Figure 2-7).

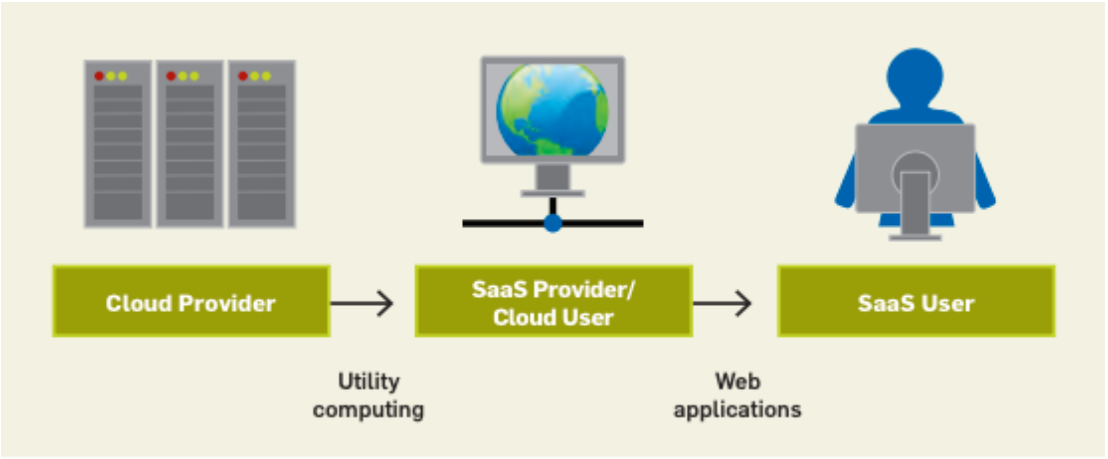


Figure 2-7: Armbrust et al. (2010) conceptual cloud to user architecture

Armbrust et al. (2010) defined ten challenges / opportunities for growth of cloud computing in the near future and recommend these as future research topics and development projects. Particularly interesting in their points is the *data lock in* challenge which reminds us of Dey's (2001) points for context-aware computing about data formats as the challenge and opportunity for communication and context-aware exchange of information.

2.5.1 Cloud computing and mobile clients

The machine environment can be different for each node in a cloud. This will again influence the context-awareness of that particular network node and thereby potentially influence the decisions made for context-aware queries. To avoid the environmental influence on context-aware cloud application performance and guarantee quality of service, it is important to develop robust techniques eliminating this effect (Mei et al., 2009).

Christensen (2009) looked into some of the issues portrayed by Mei in their work, describing a scenario for creating next generation mobile applications by utilising cloud services. He described the use of cloud technology to ease off the burden of the mobile clients. The features modern smartphones are lacking include large local storage, processing capabilities, and constant information connectivity. These can easily be supplemented from cloud computing providers and can even be further extended by including representational state transfer (REST-full) web-services interfaces for simple, easy consummation of the services. Smart mobile devices can then exploit context-awareness and sensors to create rich activity contexts in a

dynamic interaction with cloud services. This leads to a three layer architecture consisting of mobile devices, REST-based cloud computing and context sources (Christensen, 2009).

In a cloud based architecture, with possibilities for deploying to / from and to run application instances from the cloud, messaging is very important as a tool for communication as well as for data transfer. Messages need to be communicated and integrated into mobile computing for consistently delivering/pushing. Up to date information to the users can then efficiently be used. The main alternative to push messages is to use a polling mechanism. This works by having the application query the server for potential updates at given time intervals. There are several challenges of the polling approach, including deciding poll-request frequency. This is vital to mobile devices as issues like battery life and costs of network communications must be dealt with.

Push (and pull) messaging is also a described thematic area within context-aware computing (Michahelles and Samulowitz, 2002; Cheverst et al., 2001). Moreover, it is also a major part of a cloud-based architecture in terms of what messaging scheme to implement. Pull and push of information can for the first type be exemplified through user requesting information by activating a mechanism in the applications and for the second information retrieval can be triggered by different actions such as a sensor reaching a level. Push and pull of information have been explored through early context-aware research when incorporating location in the applications. Cheverst et al. (2000) investigated this in their mobile tour guide application.

Following ideas from the work of Christensen (2009) the combination of mobile computing and cloud computing are characterized in research (Khan and Ahirwar, 2011; Goyal et al., 2012; Guan et al., 2011) as mobile cloud computing. Cloud computing has established itself as a scalable, distributed and abstracted environment for Software as a Service (SaaS) solutions or the less abstracted Platform as a Service (PaaS) (Mei et al., 2008). Mobile cloud computing can be seen as an extension of cloud computing (Qureshi et al., 2011). This concept adds new clients to the solution

portfolio such as: mobile phones, tablets, and netbooks. This is an expansion from the more traditional desktop clients and servers.

Mobile cloud computing can be viewed as a cross-disciplinary area rooting both in mobile computing as well as cloud computing (Huang, 2011). Mobile cloud computing tries to address limitations found in mobile devices, as earlier mentioned in the section on mobile computing, identified in the works of (Satyanarayanan, 1996). Simoens et al. (2011) and Guan et al. (2011) as well as (Qureshi et al., 2011) also support these main challenges identified and they can be summarized as:

- Network latency and limited bandwidth
- Limited resources on the mobile phone (Battery, CPU power, storage)
- Security and privacy

In regards of network latency this challenge is faced due to the heterogeneity of the mobile networks ranging from GPRS, EDGE, 3G and 4G to WLAN. Klein et al. (2010) describes this challenge in their work, and describes three different models of communication based on limited, normal and extensive amount of bandwidth available. The mobile phones will also possibly be out of network range for shorter or longer periods, obstructing any communication and taking them offline. Entering basements and subways, visiting remote areas or on-board airplanes are just some examples of such offline situations. This puts pressure on the mobile cloud infrastructure to handle such varied forms of connections. Efforts to solve these issues was exemplified by Guan et al. (2011) in their description of a framework for wrapping mobile cloud computing applications. Here resource scheduling and separate client and cloud interfaces were measures taken towards securing connectivity and stable performance.

The issues of limited resources on the mobile phones is an important topic to solve in mobile cloud computing. Mobile phone hardware is continuously improving and mobile phone specifications of today are quite impressive compared to state-of-the-art only two years back. But, similarly, the applications of today are evolving as well demanding more resources, faster CPUs and better graphics. This maintains a gap

between mobile phone resource utilization and applications developed (Guan et al., 2011; Goyal et al., 2012). This challenge is partly solved by deploying thin clients to the mobile phones, with the user interface layer as link between the cloud application logic and user interaction (Simoens et al., 2011). The separation of business logic and data persistence from the mobile user interface is often referred to as offloading (Simoens et al., 2011; Guan et al., 2011) tasks and resources from the mobile clients.

Ultimately the goal of mobile cloud computing is to continue to improve the end user experience. By combining mobile computing, cloud computing and context-aware computing distributed resources and applications create opportunities for innovation and for enjoying applications on the move unconstrained by local limited resources and empowered by secure, scalable and flexible cloud solutions (Guan et al., 2011).

2.6 Research aims and objectives

Through the literature review, the need for using context-awareness as a main source of information to tailor information and deliver adapted user experiences has been highlighted. However, much research is related around one or two sources of such information, making the solutions somewhat limited in their behaviour. Different studies have explored context-awareness in terms of scenarios for meeting rooms (Chen et al., 2004), tourist situations (Ferris et al., 2010) or used in health care (Wojciechowski, 2009), and research by Lovett and O´Neill (2010) supported the focus on multi-source context-aware applications. Moreover Mei et al. (2008) suggested that the link between cloud computing and ubiquitous computing to be strengthened and Schwinger et al. (2005) as well as Randell (2004) agreed that on device context-aware information should be enriched with cloud aggregated data and the potential of such integrations are not yet realized.

Accordingly, I have defined the following research aim for my study:

> *"To investigate multi-dimensional context-awareness in collaborate interplay with cloud-based mobile services, in heterogeneous smartphone applications, in order to create adaptive user experiences"*

In particular, I aim to examine the implementation and exploitation of multidimensional context-awareness in mobile devices. The use of multiple sources for context-aware information has been identified as an important factor for creating adaptive solutions able to tailor information to users' needs. To ensure a full, comprehensive out of box experience evaluation I shall follow industrially designed experimental methodologies, and this work is detailed in chapter four. Accordingly, the first objective of the thesis is to:

> *Implement and integrate multidimensional context-aware functionality in a solution for a mobile personal information manager.*

The link between cloud based services and context-aware information has been highlighted as an important research subject and this shall be assessed from a practical perspective in an intelligent meeting room scenario. To this end, a cross platform context-aware application is developed and integrated with and than evaluated under multiple platforms and operating systems. My work in this respect will be described in chapter five. Accordingly, the second objective of the thesis is to:

> *Explore the integration of context-aware information from cloud-based services and applications in heterogeneous mobile applications.*

After investigating multidimensional context-awareness as well as cloud integration in a heterogeneous environment, I further seek to assess the impact of a real world, full-scale context-aware experience on a mobile information platform. By utilizing research results from multi-dimensional context-awareness and from cloud integration I intend to develop a new adaptive user interface for content tailoring on a

mobile phone, and work in this respect is detailed in chapter six. Accordingly, the third objective of the thesis is to:

> *Create an adaptive user experience by exploiting multidimensional context-aware information in relation to cloud computing on an Android device.*

2.7 Conclusion

This chapter has introduced issues concerning mobile applications and the use of context-aware information and cloud integration. Research concerning exploitation of context-awareness, as means for creating an adaptive and tailored user experience, has been reviewed. The need for more comprehensive studies and exploration through artefact development for further exploring on demand, tailored experiences and full exploitation of smartphone device functionality has been justified. The methodology chosen for the three experiments as well as for guidance of the overall PhD research project will now be presented and justified in the next chapter.

Chapter 3

Research Method

In the previous chapter, the research gap, the main aim of the research and the objectives highlighted through the literature review were presented. In this chapter, the research method used in this work is presented and justified. This includes an in-depth discussion and justification for the chosen approach for development and evaluation. The research method is applied to realize all the objectives for meeting this aim, and ultimately answer the research question. This chapter starts out by identifying the need for a research method and gives a general overview of research. Further, the chosen research perspective is presented, before I stepwise describe and justify the research methodology. The materials used in this research are then presented, before the chapter concludes.

3.1 General Approach

Research can be seen as an activity contributing to the domain knowledge within a given field. By performing it as an activity, it contributes to the understanding of a phenomena (Vaishnavi and Kuechler, 2004). The process of research often includes a systematic investigation into a problem or a domain to acquire new knowledge, establish facts, prove ideas or propose new theories. Data discovered forms the basis for supporting new knowledge and underpinning the results and conclusions. Hevner and Chatterjee (2010) characterized research by eight characteristics:

- Research originates with a question or problem
- Research requires a clearly expressed goal
- Research follows predefined process of steps
- The problem or question is usually redefined or more fine-grained

- A research problem or a hypotheses guides the process
- Research accepts some critical statements
- Data needs to be collected and interpreted
- Research is by nature cyclic or iterative

There are numerous methods that one can use in order to conduct research and this will have an impact on the process and how one should interpret the data. In the field of science, it is the goal to remove the looser meaning and acquire solid information, which will have validity only in its proper context. Scientific research adheres to a set of strict protocols and long established structures for performing the research. The chosen research perspective and methodology determines whether the findings from the research can be accepted and taken seriously. These protocols can vary slightly between scientific disciplines, but all follow the same basic structure of research aim / problem definition, literature review, design and evaluation, discussion and conclusion / communication of research. I will further go into the research perspective for this research, before presenting and justifying the chosen research methodology.

3.2 Research Perspectives

Positivism states that the goal of knowledge is to describe the phenomena that we observe and holds that the purpose of science is to grasp what we can observe and measure. Positivists build on the principle that reality can be objectively described by (measureable) properties (Myers and Avison, 2002). Positivism seeks to acquire authentic knowledge through sense, experience and positive verification. A positivist's research focuses on science as a product and results are gathered through means of measurement or quantification, hypotheses testing and inferences about a problem from a sample to a stated population (Orlikowski and Baroudi, 1991). Examples of quantitative research methods include surveys, experiments and numerical methods such as mathematical modelling. This is in contrast to qualitative methods used for studying cultural and social phenomena, and these methods include action research, case study and text/document/interview interpretation (Myers and Avison, 2002).

Contradictory to the positivist paradigm perspective, the interpretive paradigm has a different research foundation. Interpretive research assumes that reality can be studied through observation and that theory will arise from data collection. This differs from the natural science by applying a deductive theory approach (Grix, 2004). Interpretivists emphasise the subjective meaning of the reality, and how this is formed through participant interaction. They seek scientific knowledge through understanding of human and social interaction with a particular focus on the individually constructed meaning of reality. Inputs from participants are used as primary sources for new understanding of phenomena (Walsham, 1995). To achieve this, researchers based on interpretivism often interact to observe from the participants view of the world (Chen and Hirschheim, 2004). Generalization from one or a few studies are not the goal in interpretivism, rather the intention is to understand and the deeper structure of the phenomena, for then to see how this might have inferences in other situations (Orlikowski and Baroudi, 1991).

The positivist versus interpretivist perspectives can be summarized as follows (Chen and Hirschheim, 2004; Vaishnavi and Kuechler, 2004) (Table 3-1):

Table 3-1: Philosophical assumptions for the positivist versus interpretive

	Positivist	**Interpretive**
Ontology	A single reality. Knowable and probabilistic	Multiple realities. Socially constructed
Epistemology	Objective, distant, detached observer	Subjective, researcher participates and interacts
Methodology	Observation, quantitative, statistical	Participation, qualitative, explanatory

The main aim of this research is to investigate ubiquitous computing and context-awareness in a mobile environment and explore this in relation to creating tailored and adaptive user experiences. Although several approaches could be taken, empirical measurement is essential to determine the nature and frequency of social phenomena. This is in keeping with the summarization of the positivist research perspective (Orlikowski and Baroudi, 1991):

- Ontologically: Positivists believe that reality exists objectively and independently from human experiences
- Epistemologically: Positivists believe in the testability of theories. Scientific knowledge should allow verification and they seek generalizable results.
- Methodologically: Positivists take a value-free position and employ objective measurement, such as a survey, to obtain evidence in the research.

Following these lines, this research will use the quantitative measures from user evaluation in combination with measurable numbers from testing and verification techniques in software engineering to obtain objective, independent and within a given, limited context, generalizable research contributions. Investigating patterns and relations of variables in this research, interesting results could be seen emerging from the data. The interpretations of results are based on evidence that has been systematically gathered and analysed. Additionally, research can either be descriptive or explanatory (Vaus, 2001). This research addresses questions and issues so far not investigated by context-aware mobile research and the goal is to empirically investigate this topic. I will further present, detail and justify the research method applied to achieve the research aim.

3.3 Design Science and Design Science Research

With this research based on the positivist paradigm, and concerned with problem solving tasks within the fields of applied information communication technology, the research methodology will need to support software engineering and solution development as part of the means of study. As Design Science Research is fundamentally a problem-solving methodology with research based in construction and evolution of artefacts (Hevner et al., 2004) this is an approach well-suited to meeting the aims and objectives of this thesis, as I shall further elaborate and justify.

During the last ten years, Design Research has evolved and been established as a vital method for conducting research. The main focus of Design Research lies in the building and evaluation of concrete artefacts, and their evolvement. Design is to invent and create something that does not exist in nature (Vaishnavi and Kuechler, 2007) and Design Research strives to "create things that serve human purposes"

(March and Smith, 1995). Design Research can be seen as a feasible way of creating solutions which are of high relevance for business practice (Niehaves, 2007). This separates a bit from natural sciences where the goal is to understand reality.

March and Smith (1995) presented *constructs, models, methods* and *instantiations* as four types of products from Design Science Research. *Constructs* represent the initial phase of a Design Research project and is the first output, establishing the domain vocabulary. *Constructs* are thus used to conceptualize the problem domain, and are the first step towards specifying the first of series incremental steps towards a solution. Further into *models*, this represents the presentation of the relationship and association between the different models and descriptions. This can then be utilized to describe theories and knowledge of existing science, but also be used to describe relationships between *constructs*. The *model* can be represented as a description on how things are or how phenomena can be understood. Realisations of *models* can be concrete as entity-attribute-relationship (EAR) diagrams for modelling data or as a description defining a component in a design task. Through the output from *methods* the steps for undertaking the problem solving and realizing the relationship between *constructs* are described. This will often be in the manifestation as an algorithm or a guideline. The final product from the process is the *instantiation*. This is the realization of an artefact in a given environment. Instantiations provide working artefacts, and these may well be communicated as grounding for further study (March and Smith, 1995).

Within the area of Design Science Research, we find two schools of thought. On one hand we have the Design Research and on the other hand we find Design Science (Hevner et al., 2004). Cross (2001) also described similar categories but separated them into 'science of design' and 'Design Science'. I will further follow the definitions of Hevner and use the categories of Design Science and Design Research. Design Science is categorized as research conducted upon the Design Research process and aims at creating and improving the standards for its rigour. Design Research on the other hand is targeted at creating and evaluating artefacts to solve specific classes of problems (Winter, 2008). Not every artefact designed can be categorized as Design Research. To be able to do this there must be some aspect of

generalization to the results and to achieve this it requires a fine balance between being able to generalize upon a produced artefact and the level of abstraction in the problem solved. If the scenario to be solved is too generic, it will be very difficult to create reasonable artefacts that solve the problem, but the potential of generalization will be good. On the other hand, if the scope of the scenario is too fine-grained, it will be easier to create a suitable artefact to solve the scenario, but much more difficult, if possible at all, to be able to generalize from the solution. To be able to balance this matter and address the issue, situational artefacts are used. Backed by concepts from theories of management decisions by Fiedler (1964) and Simon (1947), situational artefacts aim at maintaining the ability to generalize at the same time as a situational scope is included to solve a specific scenario.

Venable (2006) separated the two into Design Science and Design Practice. He supported the argumentation of Design Science Research as a method of conveying results applicable for a class of problems to a type or class of stakeholders, whereas design practice is the low level technical solution or developed design in order to solve a specific problem in a given situation to a defined group of stakeholders. Further, Venable discussed the need for theorising as parts of the Design Science Research. Venable argued for the position of theory and theorising in Design Science Research and proposed a framework for utilising theories and ground them in technology design, knowledge of the problem area and technology evaluation (Venable, 2006). By drawing in detail upon each of these three areas, new or improved theories may be communicated.

This work is not trying to improve upon or add contributions to general acceptance and use of Design Research / Design Science Research. Rather, this research will apply the methodology to achieve a basic structure and adhere to research protocol. Therefore the definitions and perspectives from Hevner (2010) and Winter (2008) on Design Research as a framework and methodology for creating and evaluating artefacts to solve a specific class of problems will be the guiding perspective for this research.

3.3.1 Design Research

Research in the fields of Information Systems and Computer Science has long traditions in grounding the research in established methodologies. In particular, the Software Engineering research community is adapting the Design Research methodology (Vaishnavi and Kuechler, 2004) to its needs. Design Research strives to create things that serve human purposes (March and Smith, 1995), a goal which differs from natural sciences where the goal is to understand reality. Hevner et al (2004) discussed this methodology and stressed several benefits in using the Design Research approach, including continuous building and evaluation of artefacts. This is also in accordance with the statements from Vaishnavi and Kuechler (2004). Design Research is thus a well-grounded and well-founded process to use as the research methodology. Accordingly Design Research, as interpreted and implemented in this research, is a methodology building on the perspective of the earlier presented positivist views by applying quantificational measures and observation as evaluation measures. Design Research involves the use of artefacts to understand and interpret results. These artefacts can be represented as program code, software components, user interfaces or system designs. This involves a complete life cycle from design and implementation to use and evaluation.

The actual design of the research process varies somewhat between the different contributors. Gregg et al. (2001) suggested a framework for building and sustaining quality in software engineering research. Their framework consisted of a conceptual phase for theoretical grounding, followed by a phase of formalization and ending with the phase for development (Gregg et al., 2001). Vaishnavi and Kuechler (2004) suggested a similar framework for the Design Research methodology (Figure 3-1), but somewhat extended to include a more detailed process.

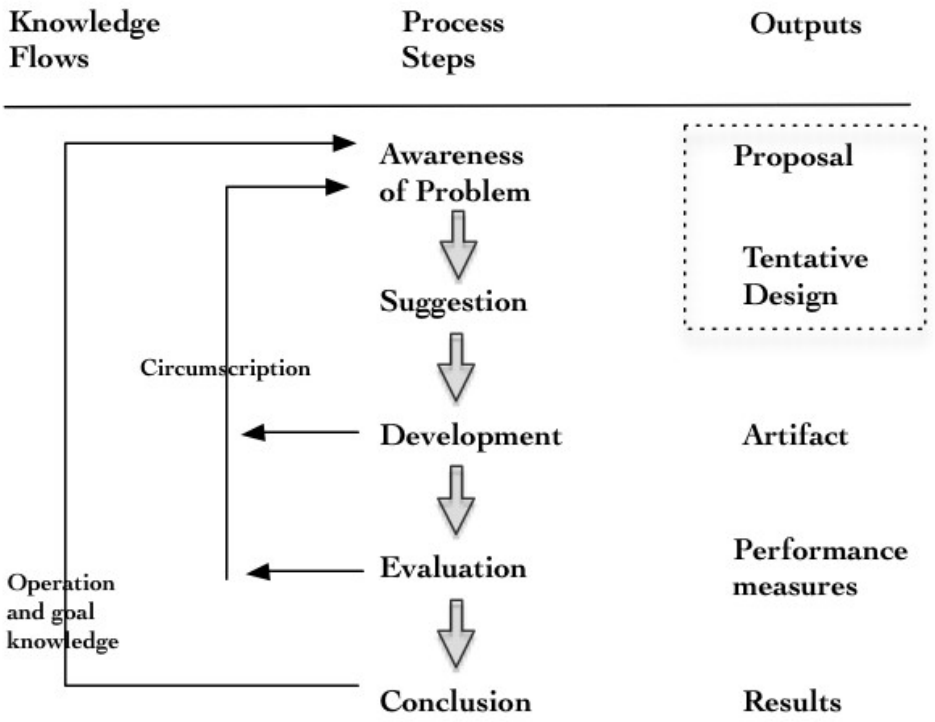


Figure 3-1: Design Research process model (Vaishnavi and Kuechler, 2004)

Even though they suggested a detailed framework, it is important to keep in mind that there will be individual variants from project to project and that these steps act only as a guideline. This figure deals with the general methodological approach and emphasizes how the research can be founded in an established methodology. Further work that has to be done as the research is starting is to create a detailed description matching the steps shown in the above-presented figure to meet the requirements and research goals.

Research from Peffers et al. (2007) has created an alternative, but somewhat similar framework, and Figure 3-2 below outlines their proposed framework.

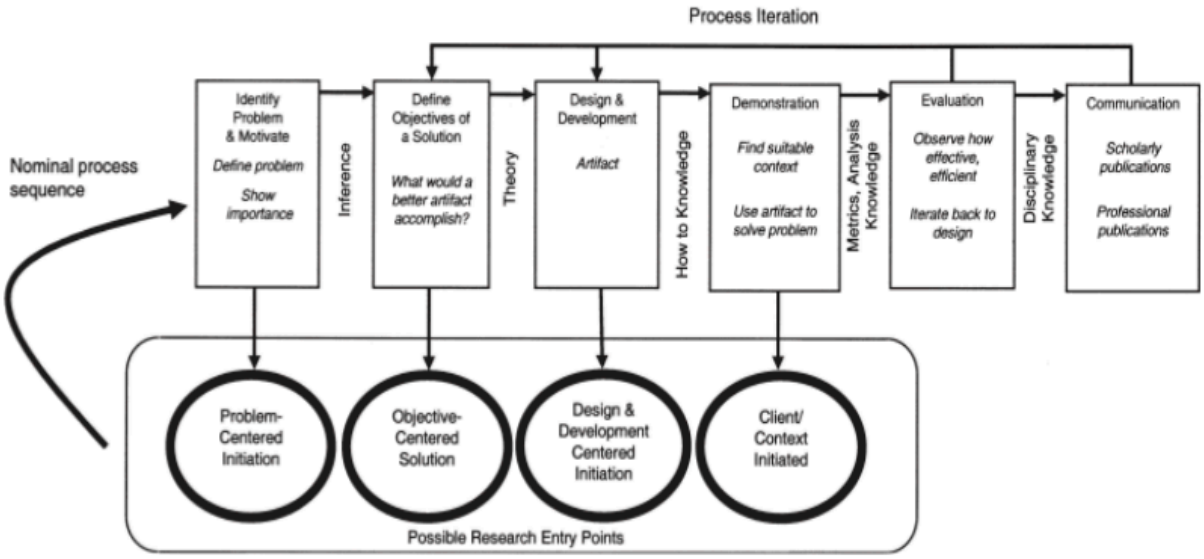


Figure 3-2: Design Research process model (Peffers et al., 2007)

As visualized in their model there are several entry points for a research process to start from. This is a main difference, separating their approach from the one described by Vaishnavi and Kuechler (2004). The different entry points do not relate to specific areas or topics of research, but rather they are related to the specific focus of the research:

- *Problem-Centred Initiation*: The origin of problem to be researched may be previous research or an observed phenomenon

- *Objective-Centred Solution*: This entry point can often be initiated by a need from industry or research to have a specific problem addressed

- *Design & Development Centred Initiation*: The existence of an artefact produced but not yet verified as addressing this research gap. Such an artefact might as well originate from a different research domain

- *Client / Context Initiated*: Applying research rigour backwards in the process model in order to address a working solution observed

3.4 Design Research and This Study

As presented in the previous section and consistent with Hevner et al. (2004) a Design Science Research project seeks a solution to a real-world problem of interest to practice and research. To be able to visualize how the Design Research process can best realize this, this research work is structured following the general approach from the work of Vaishnavi and Kuechler (2004). The model presented by Peffers et al. (2007) could also have been chosen, with the *problem-centred initiation* step as the starting point. This was considered, but since this research project starts off with roots in a business domain and from an awareness of issues in mobile computing as a research domain, the more compact model from Vaishnavi and Kuechler (2004) is chosen. Each step in Vaishnavi and Kuechler (2004) Design Science Research process is represented in the structure of this thesis. Table 3-2 describes how the Design Research framework has applicability and a mapping to each thesis chapter.

Table 3-2: Design Research applicability and mapping to thesis chapters

Design Research by Vaishnavi and Kuechler (2004)	Research process step description	Chapters mapped to research process step
Awareness of problem	Define the research domain and the challenges to be addressed	Chapter 1: Introduction
Suggestion	From the challenges identified, qualitative and quantitative objectives should be defined.	Chapter 2: Literature Review Chapter 3: Research Method
Development	Create artefacts with research contribution as a grounded part of the design. Apply case study, experimentation, simulations, tests or other appropriate activities to demonstrate how instances of the defined challenges are solved.	Chapter 4: Research experiment 1 realizes objective 1 Chapter 5: Research experiment 2 realizes objective 2 Chapter 6: Research experiment 3 realizes objective 3
Evaluation	Observe, measure and evaluate using rigours techniques and metrics. Results produced can include both quantifiable metrics and empirical evidence.	Chapter 4: Research experiment 1 realizes objective 1 Chapter 5: Research experiment 2 realizes objective 2 Chapter 6: Research experiment 3 realizes objective 3
Conclusion	The problem, objectives, the design, data and the results along with justifications for chosen approach must be communicated back to the field of research.	Chapter 7: Conclusion

I will further elaborate on this mapping by inspecting each of the steps in the Design Research process as laid out by Vaishnavi and Kuechler (2007) and describe the relevant realization of these in my research.

3.4.1 Awareness Phase

Following Vaishnavi and Kuechler (2007), the initial phase is the awareness phase. This is the pre-phase in which the understanding and problem identification has begun for the research project. The Research domain is identified and key concepts and knowledge is gathered as background information. Further concepts, ideas and business domains are recognised and exhausted for more background information to create a wide basic knowledge foundation. Refining and sorting this information leads to the definition of the problem domain and help form ideas about which questions to be answered (Vaishnavi and Kuechler, 2004). An understanding is also developed for the business domain by extracting ideas from new developments in industry.

For this research project, the awareness phase investigated current trends and issues within the domain of ubiquitous and mobile computing. By consulting the literature as well as business solutions / software solutions, an understanding of relevant issues was obtained. The results from this stage are represented in the introduction chapter and the initial concepts for the foundation of the literature review.

3.4.2 Suggestion Phase

Following, in the suggestion phase, challenges are identified and qualitative/quantitative objectives are defined to form the research aim. Theoretical work in combination with modelling and proof of concept experimentation help form the background for which this is decided (Vaishnavi and Kuechler, 2007). If, after prototyping, there is not a design ready to be presented to the researcher, then the drafts are reconsidered or the process is restarted in the awareness phase.

To realize this phase, a full literature review was conducted, as can be seen in chapter two, culminating in definition of the research aim and objectives. The methodology chapter was afterwards produced to identify suitable approaches to undertake the research problem identified. The evaluation strategies and the methodological underpinning is presented and justified. All these goals just

mentioned concern the research project as a whole. Following Design Science Research, by applying an incremental and iterative approach, these goals may be refined or detailed further in a later iteration. Moreover, this stage contains the first thoughts for the initial form of the research project. Proof-of-concept development and foundation in literature were sought to justify the research project, and this is described in chapter four, Research experiment 1: Multi-dimensional context-aware tailoring of information. The results from the evaluation of the first research experiment created the foundation for the awareness of the second research chapter, which then provided research results used as a basis for designing the third research experiment. As these build incrementally on each other, they all strengthen the use of Design Research as an overall methodology for the work undertaken in my thesis.

3.4.3 Development Phase

The development phase is very important for realizing the artefacts and constructs. This is the phase for production of the artefacts to answer the given research question. Theories, domain knowledge and conceptual modelling are realized here into concrete artefacts (Vaishnavi and Kuechler, 2004). The choice of realization will vary depending on the goal. If the artefact were an expert system, it would require software development, whereas the artefact being an algorithm may lead to construction of a formal proof instead. The novelty of the artefact lies in the design and output, and not necessarily in the construction process. Still, the legitimacy of the construction process is strengthened by employing state-of-the-art techniques from the given field (Vaishnavi and Kuechler, 2004).

The research contribution from this thesis, namely answering the aims and objectives detailed in chapter two, is a theoretical contribution to the research field. To be able to support the contribution, incremental steps were undertaken in terms of artefact production. These artefacts were produced in three individual research experiments and they are further described and explained in detail in chapters four, five and six. Together, these three research experiments undertaken in an iterative and incremental manner realize the overall research aim and create three major artefacts. The overall research contribution are presented and discussed in chapter seven. Together with the results, limitations of the work are presented and possibilities for further work are identified.

For each of the three experiments an individual Design Research approach was undertaken. Accordingly, each of these experiments applied Vaishnavi and Kuechler's (2007) five steps for Design Research: awareness of problem, suggestion, development, evaluation, conclusion. This creates two immediate benefits. Firstly, each research experiment follows an established research method and stepwise procedure for creating artefacts representing new knowledge and the output can be categorized as *constructs, models, methods* and *instantiations,* following the work of March and Smith (1995) as described earlier in this chapter. The described mapping between the Design Research process and the different research experiments and their corresponding chapters is thus given in Table 3-3.

All the three research experiments involved software development of an expert system. To be able to better control this software development process and manage the continuous relevance to the chosen problem domain, state-of-the-art software development techniques were applied. This included techniques from the eXtreme programming domain (Beck, 1999) such as small releases, tests and incremental and iterative development to support rigorous results. Iteration over design (development) and evaluation (testing) is one important difference of Design Research compared to natural science or behaviour science experimentation (Hevner and Chatterjee, 2010). To adhere to these concepts in this research project an agile approach to development and evaluation has been adopted, following the principles of Scrum (Schwaber and Beedle, 2002). This is used as the overall software-developing framework for software engineering, and the previously described eXtreme programming techniques are tools and practices incorporated into this and applied as practices in the executing parts.

Table 3-3: Design Research mapping to chapters and research experiments

Design Research by Vaishnavi and Kuechler (2004)	Individual research experiment steps	Corresponding section of research experiment chapters (please refer to chapter 4-6)
Awareness of problem	Awareness obtained through study of literature, consulting industry and, for research experiment two and three, through results from the preceding experiment	Introduction
Suggestion	In this phase the main literature were identified and preliminary design and architectures produced.	Literature review recap
Development	This phase consisted of artefact production. All the three research experiments involved construction of software in form of an expert system. The phase were structured by using incremental and iterative techniques, thus continuously improving design, architecture and implementation of the underlying constructs.	Design and architecture
Evaluation	In each research experiment adequate measures from best-practice software engineering were applied to ensure artefact quality, including testing, code review and user evaluations. The evaluation results were discussed in light of the literature.	Results and discussion
Conclusion	Following the evaluation section research experiment results were discussed and research contribution identified. Lastly conclusions were drawn and recommendations for further research made.	Conclusion

For all the three research experiments, several rounds of refining were needed to successfully complete this phase. The concept of Scrum, as illustrated below (Figure 3-3), was adapted as the structured approach for undertaking the development and evaluation phases.

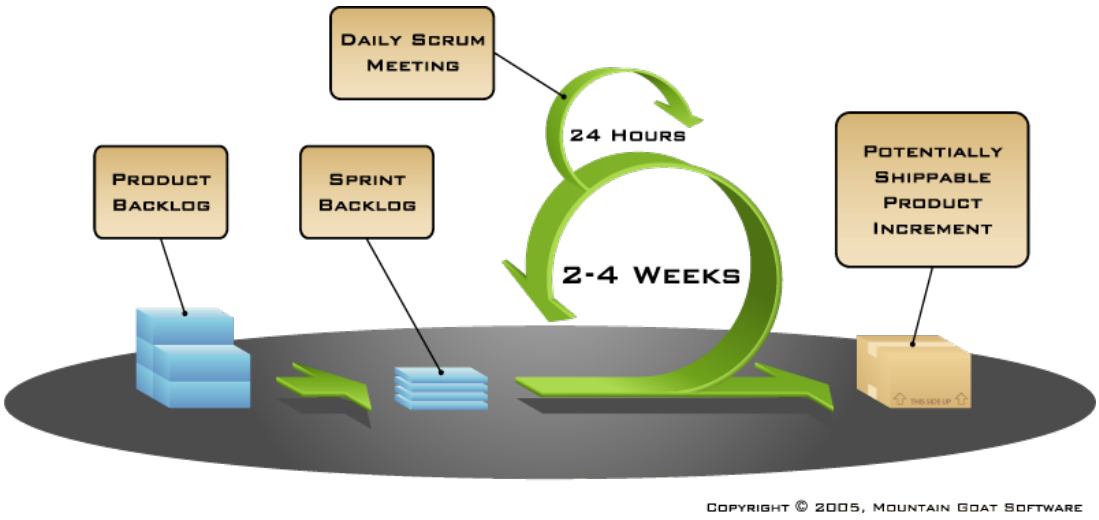


Figure 3-3: Overview of Scrum process (Scrum Alliance, n.d.)

This integrates well with Design Research as it supports the foundational idea of an iterative and incremental approach for artefact production and release. The functional specification developed in the design phase created the initial product backlog document used as foundation for task planning in Scrum. Through iterations this was improved upon, and software evaluation was performed through testing and verification. This is in accordance with theory from Vaishnavi and Kuechler (2007), stating that several rounds of development and evaluation are needed to complete artefact production. Using Scrum helped control the artefact production and maintain focus on research output, whilst adhering to state-of-the-art techniques from the business domain.

3.4.4 Evaluation Phase

In the evaluation phase, a Design Research process frequently alternates between the development phase and the evaluation phase rather than flowing in a waterfall fashion (Kuechler and Vaishnavi, 2008). Iteration between design (development) and evaluation (research experiment testing) is one important difference of Design Science Research compared to natural science or behaviour science experimentation. In this research, the agile approach of Scrum was undertaken for the development

and evaluation execution, as described previously. While Peffers et al. (2007) and Vaishnavi and Kuechler (2004) have created frameworks for structuring a Design Research process, the approaches themselves lack methods for evaluating the results of the research. To apply rigorous evaluation, the Design Research methodology was combined with guidelines from Hevner (2004) for research-based evaluation. These proposed guidelines will therefore be used as the framework for evaluating the research output of this project. The following table projects Hevner's guidelines with a description of each evaluation step (Table 3-4).

Table 3-4: Hevner's guidelines for Design Research evaluation

	Research Guideline	Description
1	Design as an artefact	In the project, realistic artefacts must be created as outcome / deliverables I.e. models, instantiation
2	Problem Relevance	The solution needs to be grounded in a business problem
3	Design Evaluation	The artefacts need to be evaluated with appropriate techniques
4	Research Contributions	The outcome of the project must add value to the existing knowledge base
5	Research Rigor	Thorough methods need to be applied both in construction and evaluation
6	Design as a Search Process	Appropriate methods in the design process must be chosen to achieve desired ends
7	Communication of Research	Project outcome must be presented clearly to all expected audience

The detailed results from the software engineering evaluation as well as from the methodological evaluation are presented and discussed in detail in the different experiment chapters (Chapter 4-6). Subsequently in chapter seven, these results and evaluations are discussed in relation to the overall theme of the research project.

3.4.5 Research validity

Following Leary (1995), internal validity is a reference to how well a research study is conducted in terms of research design, control of variables and measurement of

variables. Whereas external validity is the extent to which results can be generalized and made applicable to other cases, samples, research settings and procedures. To keep a high internal validity, control of the execution can be increased. This would mean more detailed control over the study and an increased strength in conclusions drawn from effects of the variables. The drawback from this is that the result produced might only be applicable in a particular context due to its high level of detail. This means that generalizability is decreasing and thereby lowering the external validity. By using the guidelines from Hevner et al. (2004) for evaluation, I ensure stronger external validity by using established criteria for conducting the evaluation. In this study, the research experiments undertaken are performed both in a real-world context, and in a contained, lab based, scenario. In the real-world scenario one has less control of the variables. Factors such as environment, people's behaviour, previous experience, time of the day etc. may influence our results. For the lab-based scenario, in which man have full control of the variables used, there are less external disturbing elements, but the context of the situation of use may feel artificial for the test-candidates. In experimental research the external validity can be only assessed by replication (Leary, 1995). Thus, even if the findings of this study lead to strongly correlated and clear results, there will always be a need for further research, to replicate the findings from this study in order for them to be considered generalized.

3.4.6 Research rigour

Research rigour can be viewed from several different perspectives, from which I will delve into the two most important ones. The first one is the perspective of research rigour on how to do Design Research. This is research concerning improvements of the research method itself and building theories on how to further improve upon the standards for its use. Although this is a very important aspect within the field of Design Research, this is out of scope and not relevant for this research. I focus on the second perspective which deals with the ideas and concepts of how Design Research can be conducted in a rigorous and relevant manner (Niehaves, 2007). In this work, anchoring in well-established methodology and guidelines ensures this. This leads to production of artefacts relevant to a business domain, and additionally still applicable to be generalized upon and to draw new knowledge from, all in accordance with the step-wise process by Vaishnavi and Kuechler (2004). Moreover, the guidelines by

Hevner (2004) as described earlier, are used to carry out a systematic evaluation of the research. This supports the research outcome and helps to positively increase research rigour.

3.4.7 Research experiment Questionnaires

In this study, questionnaires to evaluate the user mobile information access experience were employed at the end of each experiment. For all questions, the users had to express their opinions on a Likert scale (Johns, 2010), where the given answer reflects their position on the given dimension. Likert-type scales are often frequently used within the fields of human computer interaction, and are well suited for smaller sample sizes ($n < 50$) (Kaptein et al., 2010). The Likert scale implemented in this research contained an even number of possible responses, four categories or six categories. This compression from the most typical 5-point or 7-point Likert scale eliminates the neutral option in the questionnaires, forcing the users to take a stand to the questions and not act neutral (Kaptein et al., 2010). This shorter scale has the advantage that the shades of agreement do not become as hard for survey designers to express or as hard for respondents to distinguish and answer. Johns (2010) points out that there are no strict rules to the number of categories in a Likert-scale, but that any amount from two up to eleven can be found regularly. All questionnaires contained an equal amount of positive and negative statements to avoid bias on either side. They all contained one or more open-ended question as well, where the participants were free to express and comment upon issues they might have encountered. All questionnaires had their set of questions presented in a randomized order at the time they were given to the users to avoid response from an earlier question having impact on subsequent answers. The words used when composing the questions are actively conveyed so that they in no positive or negative fashion can influence the results given. For all questionnaires, pilot studies were performed initially with a small number of users to ensure the suitability of the questionnaire. Feedback from these pilot tests was brought back into the development of the final version of the questionnaire. All questionnaires used in the different research experiments are highlighted in the different research experiment chapters as well as available in the appendix.

3.4.8 User Experience Questionnaire

In the previous section, the overall structure and reasoning behind the questionnaires used in the different research experiments were presented. Further, I will delve with the user experience questionnaire, which was used for the broad classification of the users before they answered the detailed questionnaire in each research experiment. When issuing questionnaires to the users in each of the experiments, one were also interested to investigate whether user computer proficiency had any bearing on their experiences. The following questionnaire was used (Table 3-5).

Table 3-5: User computer proficiency questionnaire

Computer knowledge / searching		
Do you regularly use electronic mail (e-mail)?	Yes □	No □
Do you regularly use search engines (Google, Yahoo)?	Yes □	No □
Do you regularly use spread sheet applications (Ms Excel, Lotus, Open Office Writer)?	Yes □	No □
What does the use of "*" (star) signify when used in search engines? (Mention orally)	Yes □	No □
Have you ever successfully installed software on a computer?	Yes □	No □
Have you ever written and successfully run a computer program?	Yes □	No □
What does the use of " " (quotes) signify when used in search engines? (Mention orally)	Yes □	No □
What does the use of "-" (minus) signify when used in search engines? (Mention orally)	Yes □	No □

In this research, following the taxonomy of McMurtrey (2001), computer proficiency was assessed via a questionnaire that grouped participants according to their

computer proficiency in three main groups – novice, intermediate, and expert. This categorisation was based on participants completing a short questionnaire (prior to beginning the research experiments detailed questionnaire) that assessed their computer abilities and experience (Table 3-5). Accordingly, users who answered 'yes' to 0-2 of the questions were categorised as novice, those who answered 'yes' to 3-4 of the questions were deemed to be intermediate, whilst those who answered 'yes' to 5-8 questions were assigned to the expert category.

3.4.9 Analysis of Results

To analyse the results of the three questionnaires statistically, I used SPSS (Statistical Package for the Social Sciences – versions 17 and 18), which is a data management and analysis software developed by SPSS, Inc. in Chicago, Illinois, now owned by IBM (Pallant, 2011). Among its features are modules for statistical data analysis, including Descriptive statistics (Frequencies, cross-tabulation, Descriptives), Bivariate statistics (Means, t-test, ANOVA) and Prediction (Linear regression, factor analysis). SPSS is particularly well suited to survey-based (questionnaire-based) research, which is the main reason for using it in the analysis. I used a variety of data analysis techniques available from SPSS including: Frequencies, means, standard deviation, ANOVA and Factor Analysis. In the statistical analysis, the results were considered to be significant if $p<0.05$. The choice of significance level is based upon established traditional levels (Pallant, 2011). Ensuring test-retest correlation has assessed reliability of the data and validity of the scales has been an important discussion and scales sampled from the domain have adequately enforced content validity.

3.4.10 Conclusion Phase

This last and final phase of the research experiment involves presentation and conveying the research contributions. This involves summarizing and write-up of the results for public presentation to the research community. Both practitioners as well as researchers are target groups for these results and they will have to be tailored according to the recipient. Irrespective of the results, this phase can have several purposes and results will either be judged as 'firm' or as 'loose ends' (Vaishnavi and Kuechler, 2007). If the artefact produced and results measured are of such standards that they meet the hypothetical predictions, they can be written up and presented as

such. If they fail to answer the hypothesis, then they might be ruled to be not sufficient or good enough to answer the research aim and objectives. If that is a case it might be transferred back to an earlier stage of the process for refining of the problem or continuing the work. In this phase, results defined as abnormal may be credited as subjects for further research (Vaishnavi and Kuechler, 2004).

3.5 Tools, Technologies and Platforms

In this section the different platforms and device chosen for realization of the aim and objectives through the described methodology are described and justified. The devices range from feature phones, to early smartphone era devices with personal information manager capabilities, to state-of-the-art multi core smartphones of today. Devices also include traditional laptop for software development and cloud-based backend hardware, from vendors like Google. Three primary choices were behind the selection of the different hardware:

- *Availability*: What laptops, phones, smart device were available in market at the given time of project execution. This was an important factor in order to get the latest available devices.

- *Capability*: What technical capabilities were the devices capable of and what were the hardware specification and the given time of project execution. This was particularly important in order to be able to exploit technical hardware in the research experiments.

- *Platform*: What were the running operating systems of the devices. It was important to have a broad variety of operating systems in order to secure a heterogeneous test suite for the applications developed.

Hence forth, the major operating systems from the devices used are described, and then the rest of the chapter provides detailed technical specifications of test devices and justifies their selection.

3.5.1 The JavaME Platform

Java has a vision of supporting multiple platforms and devices with their philosophy of "write once, run everywhere". In 2000 Sun Microsystems reduced and adapted the standard Java platform into a small and compact version intended for running on resource constrained mobile devices. The idea was that byte code generated in the compilation process should be able to run on a broad range of devices. JavaME includes functionality and is a platform for resource-constrained device, for instance implementing the connected limited device configuration (CLDC 1.1 / 2.1). Later, Sun was acquired by Oracle, but JavaME is still a major part of the feature phone market. These devices are recognized by the fact that all features of these phones are smaller than in smartphones: smaller screens, less memory, slower processors and a lower price (Oracle, n.d.). Java is currently running on over 3 billion phones worldwide, and it provides a flexible environment for applications running on large amount of devices, besides mobile phones, including: TV set-top boxes, e-readers, Blu-Ray players and printers, to name but a few.

The JavaME platform comes in forms of an additional Java software development kit (SDK) and can be developed using all the major IDEs such as Intellij, Netbeans and Eclipse. JavaME also supports the development of graphical user interfaces through a custom made toolkit, the Light Weight User Interface Toolkit (LWUIT). Through this toolkit JavaME gets several state-of-the-art features such as expanded touch screen capabilities, graphical rendering and modern UI components (Oracle, n.d.).

3.5.2 The iOS Platform

iOS is the operating system delivered from Apple Inc. for their mobile devices including iPhone, iPad and iPod Touch. The iOS was derived from Mac OS X and it has been adapted to be compact and efficient to be able to run and take advantage of the features of different devices (Apple Inc, 2012a). A framework for programming the user interaction is included, the Cocoa Touch. It was developed and improved upon from the Cocoa for the Mac desktop technology, re-designed with features like multi-touch support, drag-and-drop design and core animation. It uses a Model-View-Controller pattern that makes it easy to wire up the UI code with the

application logic. The software development is done with the proprietary Xcode tools from Apple and the programming language for the iOS is Objective-C.

The iOS platform is backed by a major ecosystem and is the most popular application store of the smartphone market if one judges by the number of applications downloaded, which just recently passed 25 billion downloaded apps (Apple Inc, 2012d).

3.5.3 The Android Platform

In November of 2007, under the Open Handset Alliance (2007), Google released the Android operating system. The Open Handset Alliance is a group of hardware and software companies working towards the goal of having a more open mobile phone environment (DiMarzio, 2008). The Android operating system is one of the fastest growing mobile operating systems and has an impressive growth in device activations (Figure 3-4), peeking more than 500 000 devices being activated each day in Q2 2011.They are ranked by the Gartner group (2011b) to be the largest smartphone operating system by 2012, with an estimated market share of 49% of the total smartphone segment.

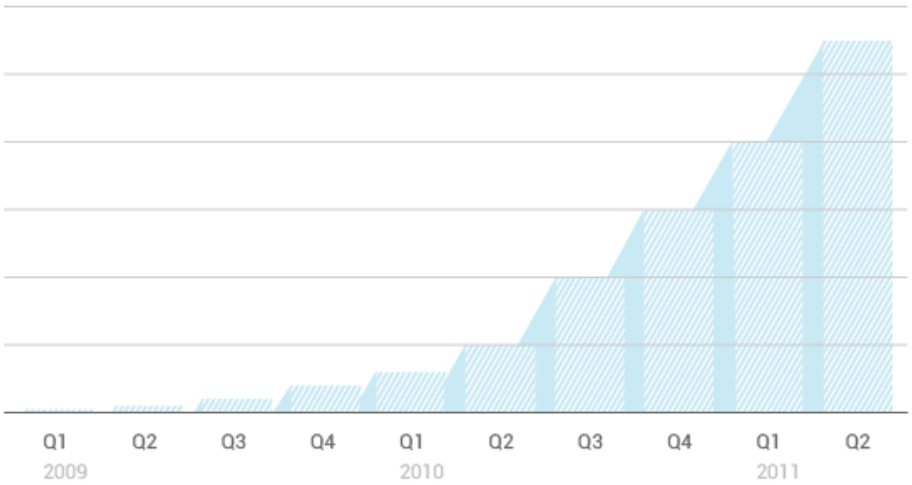


Figure 3-4: Android device activations (Android, 2012a)

Android is more than an operating system for mobile phones. Today it includes operating systems for laptops, mobile phones, TVs and servers. The latest mobile version of the operating system version 4.0, codename Ice Cream Sandwich, includes

state-of-the-art hardware and software including voice control and near field communication (NFC) capabilities (Android, 2012a).

The Android operating system is based on the Linux kernel and supports custom applications that are written in Java source code. Android comes with an application framework which supports reuse and replacement of components. The source code is not tied to the Java Virtual Machine (JVM) like standard Java, but runs instead on the Dalvik Virtual Machine, which represents the Android runtime environment (Android, 2012b). One of the main advantages of the Android platform is that the programming language and tools will be familiar to traditional Java programmers. The main tooling supplied by Google is developed for the Eclipse IDE (Android, 2011). The developer environment is a rich ecosystem including a device emulator, tools for debugging, memory and performance profiling. By offering an open development platform, Android gives developers the ability to build feature rich and innovative applications and to deploy them to the Google Play ecosystem.

3.5.4 The Windows Mobile Platform

Windows Mobile is a mobile operating system developed and shipped by Microsoft Corporation. It is based on the Windows CE 5.x series and is used as an operating system for a variety of PDAs, Smartphones and professional mobile touch screen devices (Microsoft, 2005). The Windows Mobile operating system, 6.5.x, facilitates creating custom written applications in both managed (Visual C# / Visual Basic .Net) and native (Visual C++) code. This is made possible with application programming interfaces (APIs), which have an adapted subset of the Windows desktop version. Additionally, Windows Mobile is made extensible by third party programming libraries that expose rich features as well as a programmable layer for hooking custom written modules into. The libraries constituting the Compact Framework are subset of the full .Net Framework, with some additional libraries specific for mobile development. The Compact Framework is a rich application environment facilitating for applications and databases to run locally on the mobile device. The .Net Compact framework runs with access to almost all operating system features on the device. The .Net compact framework also shares a lot of abilities with its full flex desktop version such as a Common Language Runtime (CLR). The CLR is responsible for performing the Just-In-Time (JIT) compilation on the devices for conversion of byte

code executable programme code to be used at runtime. The .Net Compact Framework uses the standard Microsoft development suite, the Visual Studio IDE, for development, debugging, testing and deployment (Microsoft, 2005).

The next generation Windows mobile operating system from Microsoft has been developed and recently shipped under the name Windows Phone. This shift marks a new branch in the history of these devices and significant changes to the platform have been made. The Windows Phone 7 is not backward compatible and neither are the 6.5.x devices upgradable to the new generation. This new generation operating system has been rebuilt and supports more modern technologies such as Silverlight, Windows presentation foundation and XNA. An eco system is launched in connection to this and this *Windows Phone Marketplace* are Microsoft's answer to Apple App Store and Android's Google Play (Microsoft, 2012b).

3.5.5 Symbian Operating System

The Symbian operating system is used in several devices shipped by Nokia. Symbian is used as the operating system in a large variety of both feature phones and smartphones (Nokia, 2012b). The feature-rich, and customizable operating system include technologies such as NFC, and environments for business-, gaming- and music applications. Symbian based devices support the development of applications through the use of Symbian C++, Qt development SDK or Java through MIDP 2.1. As expected for native application development, connectors for using multimedia phone features and native data sources such as the calendar are included (Nokia, 2012b).

3.5.6 Experimental Devices

In this section, the detailed information about the technical specification and the selection criteria of each one of the devices used in the experiments are presented. The devices vary from traditional laptop computers, personal digital assistants (PDA) to second-generation smartphones through to large display tablet devices.

3.5.7 HTC P3600 Windows Mobile

This device (HTC, 2012) was one of the first personal information managers and personal digital assistants with full phone capabilities. The HTC P3600 (Figure 3-5) has a 2.8-inch TFT resistive touch sensitive display supporting 65 K colours with a screen resolution of 240 x 320 pixels. It runs Microsoft Windows Mobile 5.0 for Pocket PC operating system on a Samsung SC3 400 MHz processor. Further, it comes with several options for wireless connectivity including WI-FI 802.11b/g, Bluetooth and GPRS/EDGE. The default kit includes 64 MB standard memory and 128 MB internal flash ROM; the memory can be extended through the use of miniSD cards.

Figure 3-5 HTC P3600 (HTC, 2012)

3.5.8 HTC TYTN II

This personal information manager (HTC, 2012) comes with full phone capabilities as well as representing a full-scale mobile office with the support for writing documents, preparing presentations and reading spreadsheets on the go. The HTC TYTN II (Figure 3-6) has a 2.8-inch TFT resistive touch sensitive display supporting 65 K colours with a screen resolution of 240 x 320 pixels. It runs Microsoft Windows Mobile 6.0 Professional operating system on a Qualcomm MSM7200 400 MHz processor. Moreover, it comes with several options for wireless connectivity including WI-FI 802.11b/g, Bluetooth and 3G/GPRS/EDGE. The default kit includes 128 MB standard memory and 256 MB internal flash ROM, and the memory can also be extended through the use of microSD cards. This device also comes with a full-scale keyboard for easier interaction.

Figure 3-6 HTC TYTN II (HTC, 2012)

3.5.9 Nokia E61

The Nokia E61 is a third edition smartphone device (Nokia, 2012a) from the E-series range, featuring a full QWERTY keyboard and targets business users in the European market. The Nokia E61 (Figure 3-7) has a 2.9-inch TFT display supporting 16M colours with a screen resolution of 320 x 240 pixels. It runs the Symbian 9.1 operating system on a 220 MHz Dual ARM 9 processor. Further, it comes with several options for wireless connectivity including WI-FI 802.11i/e/g, Bluetooth and GSM. Default kit includes 64 MB standard memory and 64 MB internal flash ROM, memory can be extended through the use of miniSD cards. This device also comes with a full-scale keyboard for easier interaction.

Figure 3-7: Nokia E61 (Nokia, 2012a)

3.5.10 HTC Magic

The HTC Magic (Figure 3-8) is an early available Android phone (HTC, 2012), featuring sensors such as an accelerometer and compass as well as no-keyboard set-up, but with track ball for interaction. The HTC Magic features a 3.2-inch TFT capacitive touchscreen display supporting 65K colours with a screen resolution of 320 x 480 pixels. It runs the Google Android operating system on a Qualcomm 528MHz ARM 11 processor. Furthermore, it comes with several options for wireless connectivity including WI-FI 802.11b/g, HSPDA, HSUPA, Bluetooth and CDMA 800/900/1800/1900. Default kit includes 288 MB standard memory and 512 MB internal flash ROM.

Figure 3-8: HTC Magic (HTC, 2012)

3.5.11 HTC Google Nexus One

The HTC manufactured (HTC, 2012) Nexus One (Figure 3-9), shipped by Google Inc., represents one of the first world wide available commercial Android phones. The Nexus One features dynamic voice suppression and has a 3.7-inch AMOLED touch sensitive display supporting 16M colours with a WVGA screen resolution of 800 x 480 pixels. It runs the Google Android operating system on a Qualcomm 1 GHz Snapdragon processor. Additionally, it comes with several options for wireless connectivity including WI-FI 802.11a/b/g, Bluetooth and EDGE/3G/HSPDA. The default kit includes 512 MB standard memory, 512 MB internal flash ROM and 4 GB internal storage, which also can be extended through the use of microSD card. The Nexus One also comes with a host of popular Google applications, including Gmail, Google Voice and Google Maps Navigation.

Figure 3-9: HTC Google Nexus One (HTC, 2012)

3.5.12 HTC Evo 4G

The HTC Evo 4G (Figure 3-10) is an Android phone (HTC, 2012) shipped by the Sprint operator for the American CDMA network. The HTC Evo 4G features a 4.3-inch TFT capacitive touchscreen display supporting 64K colours with a screen resolution of 480 x 800 pixels. It runs the Google Android operating system on a Qualcomm 1 GHz Scorpion processor. Furthermore, it comes with several options for wireless connectivity including WI-FI 802.11b/g, WIMAX 802.16e, Bluetooth and CDMA 800/1900/2000. The default kit includes 512 MB standard memory, 1 GB internal flash ROM and 8 GB internal storage, which also can be extended through the use of microSD card.

Figure 3-10: HTC Evo 4G (HTC, 2012)

3.5.13 Samsung Galaxy Tab 10.1

The Samsung Galaxy Tab 10.1 (Figure 3-11) is the first 10.1-inch Android tablet (Samsung, 2012). The Samsung Galaxy Tab features a 10.1-inch PLS TFT capacitive touchscreen display supporting 16M colours with a screen resolution of 800 x 1280 pixels. It runs the Google Android operating system for tablets on a Dual-core 1 GHz processor. Additionally, it comes with several options for wireless connectivity including WI-FI 802.11a/b/g/n, WIFI hotspot, Bluetooth and CDMA 800/1900/2000. The default kit includes 1 GB standard memory and 16/32/64 GB internal storage.

Figure 3-11: Samsung Galaxy Tab 10.1 (Samsung, 2012)

3.5.14 Apple iPhone 2G

This device (Apple Inc, 2012b) was the first smartphone from Apple in the iPhone series. The Apple iPhone 2G (Figure 3-13) has a 3.5-inch TFT capacitive touchscreen display supporting 16M colours with a screen resolution of 320 x 480 pixels. It runs the Apple iOS operating system on a 412 MHz ARM 11 processor. Furthermore it comes with several options for wireless connectivity including WI-FI 802.11b/g, Bluetooth and GSM 2G network. The default kit includes 4/8/16 GB internal storage.

Figure 3-12: Apple iPhone 2G (Apple Inc, 2012b)

3.5.15 Apple iPhone 3G

This device (Apple Inc, 2012b) was the second smartphone from Apple in the iPhone series and its capabilities and inclusion of App Store revolutionised the personal smartphone market world wide (Apple Inc, 2012d). The Apple iPhone 3G (Figure 3-13) has a 3.5-inch TFT capacitive touchscreen display supporting 16M colours with a screen resolution of 320 x 480 pixels. It runs the Apple iOS operating system on 412 MHz ARM 11 processor. Additionally, it comes with several options for wireless connectivity including WI-FI 802.11b/g, Bluetooth and EDGE/3G. The default kit includes 128 MB standard memory and 8 GB / 16 GB internal storage.

Figure 3-13: Apple iPhone 3G (Apple Inc, 2012b)

3.5.16 Apple MacBook Pro i5

The computer (Apple Inc, 2012c) applied as the development machine in the experiments is a powerful Apple MacBook Pro 15-inch running the Apple Mac Snow Leopard (later upgraded to Lion) operating system. It features a 2.53 GHz Intel Dual core i5 processor, 8 GB memory, 500 GB disk space, 15-inch LCD display and NVidia GeForce 256 MB graphics memory (Figure 3-14). The laptop served as a local server instance and was used as the main tool for software development.

Figure 3-14: Apple MacBook Pro (Apple Inc, 2012c)

3.5.17 Google App Engine

The Google App Engine makes it possible to run web applications on the Google infrastructure. App Engine applications are easy to build, maintain and scale (Google, 2012). The Google App Engine is a cloud-based platform as a service (PaaS); the platform is pre-configured by Google and provides a much higher abstraction than an Infrastructure as a Service (IaaS) platform such as the Amazon elastic cloud. The App Engine is well integrated with Google Apps services like docs, Gmail and authentication. The platform imposes certain limitations on the developers, for example no threading API, but by enforcing these Google is able to provide very high scalability. The Google App Engine supports runtime environments for Java, Python and Go. Each environment provides standard protocols and common technologies for web application development (Google, 2012). The Google App Engine has been used as the main server installation for this research and has proven to be a stable environment for deploying the Java server

applications without having to consider hardware requirements and software installations.

3.5.18 Device to Experiment Mapping

For convenience, Table 3-6 show the mapping between devices and experiments.

Table 3-6: Device per experiment mapping

Device	Experiment 1	Experiment 2	Experiment 3
Apple iPhone 2G	✓	✓	
Apple iPhone 3G		✓	
Nokia E61		✓	
HTC P3600	✓		
HTC TYTN II	✓	✓	
HTC Magic		✓	
HTC Nexus One			✓
HTC Evo			✓
Galaxy Tab 10.1			✓
Apple MacBook	✓	✓	✓
Google App Engine		✓	✓

3.6 Conclusion

This chapter has presented the methodological approach used in this research. Theories for the different schools of thought have been presented and the process for carrying out the research has been described and justified. Design Research has been discussed in detail and the stepwise realization of this adapted for this research has been presented and justified. Additionally, experimental questionnaires for evaluation were discussed and method for result analysis reviewed. Material for experiments, including platforms, mobile phones, tablet and laptop, were presented. In the upcoming chapters four, five and six, the three major research experiments will be presented, examined and discussed. These are, correspondingly, tailoring of personal information in a PIM device, intelligent meeting room scenario and the context-aware phone.

Chapter 4

Tailoring of Context-Aware Information

4.1 Introduction

This chapter presents and discusses experiment number one, conducted as part of completion of this thesis. This fulfils objective number one, as described in chapter two, literature review:

Implement and integrate multidimensional context-aware functionality in a solution for a mobile personal information manager.

The purpose of this is to examine the implementation and exploitation of multidimensional context-awareness in mobile devices. The use of multiple sources for context-aware information has been identified in the literature review, as an important factor for creating adaptive solutions able to tailor information to users needs. In order to fulfil the objective several tasks are accomplished:

- Define sources for context-aware information
- Create an application for a PIM device
- Integrate the identified context-aware sources into the application
- Perform user evaluation to explore solution feasibility

4.2 Literature Synopsis

Dey (2001) broadened the field of context-awareness by his definition of the term context, building on the earlier definitions, stating that: “Context is any information

that can be used to characterize the situation of an entity." Dey (2001) suggested that a developer aware of the current context would more easily be able to determine the requirements and functionality of the application under development. So far, most researchers identify context on the basis of one or a few factors. This is an interesting approach, and towards the goal of generalization and general user acceptance of the solution it is an important aspect. The major player in creating the user experience is the developer, and it is fundamental that their understanding of how to exploit context-awareness is broad, thorough and is central in the development process. Another approach of user interaction is described by Prekop and Burnett (2003) in their work where they look at the cognitive elements of context. They argued for the development of context-aware applications supporting cognitive activities. They proposed an activity-centred context process and described this on a conceptual level. What is very interesting about their approach is the activity-focused approach that also acknowledges that an activity can be surrounded by context from the environment. Although they do not propose how to solve this technically they do agree that there are difficulties in defining and extracting environment information. Another application utilizing context on a mobile phone, though not a PIM device, is exemplified by Schmidt et al. (2001). They produced an application where users through a dynamic WAP interface could see the current state of their contacts before calling them. This meant all participants had to share their current status, but would benefit by not having anyone calling them if they were unavailable or busy. This context sensitive phone book was found useful by a large majority of their test group. One issue with the application is that although it was implemented on a mobile phone, it still needed a server application to control and manage the context (the users' status). Further improvement could be to transfer the control and awareness of the context to the user's device.

One example of an application taking context into account is the real estate search program developed by Zhou et al. (2006) at the IBM research centre. The users' sequential interaction history and his interests form the basis upon which the next step of search for information is built. This was implemented and tested as a desktop application. Others have looked at context-aware implementation used in mobile applications. A rather different approach to a context-aware mobile phone

application was the experiment conducted by Häkkilä and Mäntyjärvi (2004) from the Finnish telecom industry, who created a location sensitive system, which delivered multimedia messages to the user's mobile phone every time s/he approached special locations on a determined route. With location as context-aware input the deliverable to the user consisted of text and / or image with information relevant to the current user context.

Through these revisited examples and definitions it is shown that context-awareness is a common part of mobile application. This is also supported by findings in the literature review (Chapter 2), highlighting the importance of context as a major factor for well received applications leading to good user experiences. The observation to be made through the literature and as emphasized in the examples revisited here is the lack of multi-dimensional context-aware applications. The majority of context-aware examples utilize only a small portion of the available contexts and there are also openings for a stronger integration into the mobile devices. Accordingly in this chapter, an experiment for three-dimensional context-aware tailoring of information is presented.

4.3 Experiment Scenario

Peter is approaching the railway station and upon closing in on the location the system automatically verifies his position. He is placed in city zone number two, close to the railway station in the city centre. Thereafter the system communicates with Microsoft Outlook on his mobile phone and looks up what kind of activities he has scheduled. The activity has a social context meta-tag and this is extracted as well. Based on the acquired activity information, the social context and his current location, the system computes the user-context. For Peter a match is calculated and the action taken from the system is to automatically set him in contact with the ticket reservation office at the stadium located next to the railway station, where he is planning to watch tonight's match with a friend.

4.4 Application Design

The developed Personal Information Manager (PIM) application consists of three different main modules: social context module, location module and activity / task module. Each of the three modules represents one contextual dimension of information and provides separately valuable information. Application logic implemented in the software application deployed to the PIM device handles connectivity with the different modules and computes the user context. Based on this the application will decide whether or not to take action and what kind of action is appropriate.

When the inputs from each of the three modules are merged, this creates a unique foundation for the application to act further upon. When calculations are performed on this data the output is interpreted as the *user context*. One user context will represent one specific state of context of the user for a given time and a given situation. The application will on this basis present the user with tailored information. The process is depicted in Figure 4-1.

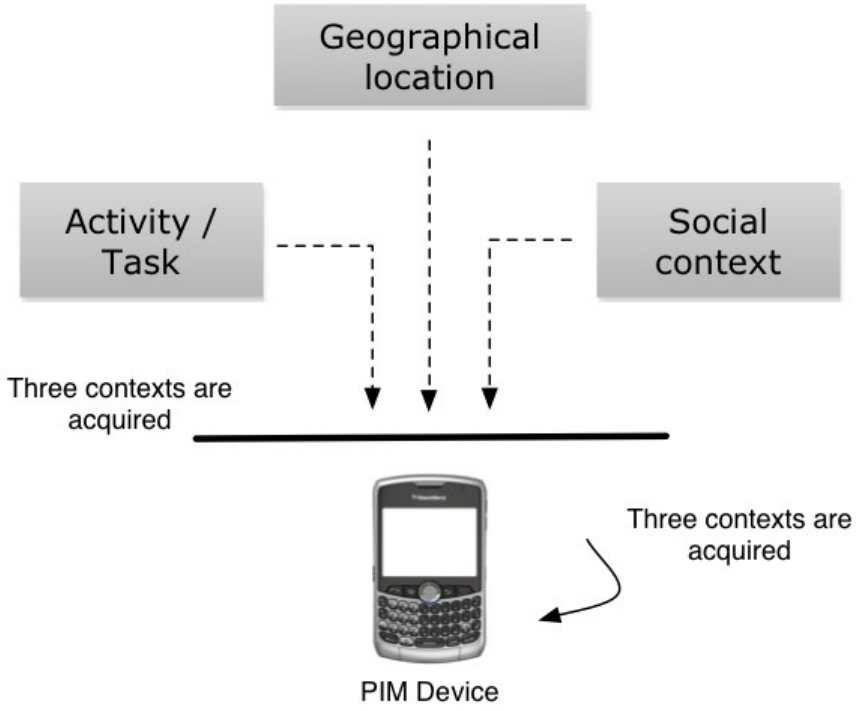


Figure 4-1: Context-aware tailoring of information from multiple dimensions

4.4.1 Application Architecture

The application was developed using Microsoft .Net C# and implemented on a Windows Mobile 5.0 device (Microsoft, 2005). The architecture of the applications follows the guidelines of Microsoft for separation of concerns and dividing the application into different classes in different namespaces. The following figure (Figure 4-2) shows the main architectural components of the application and their associations.

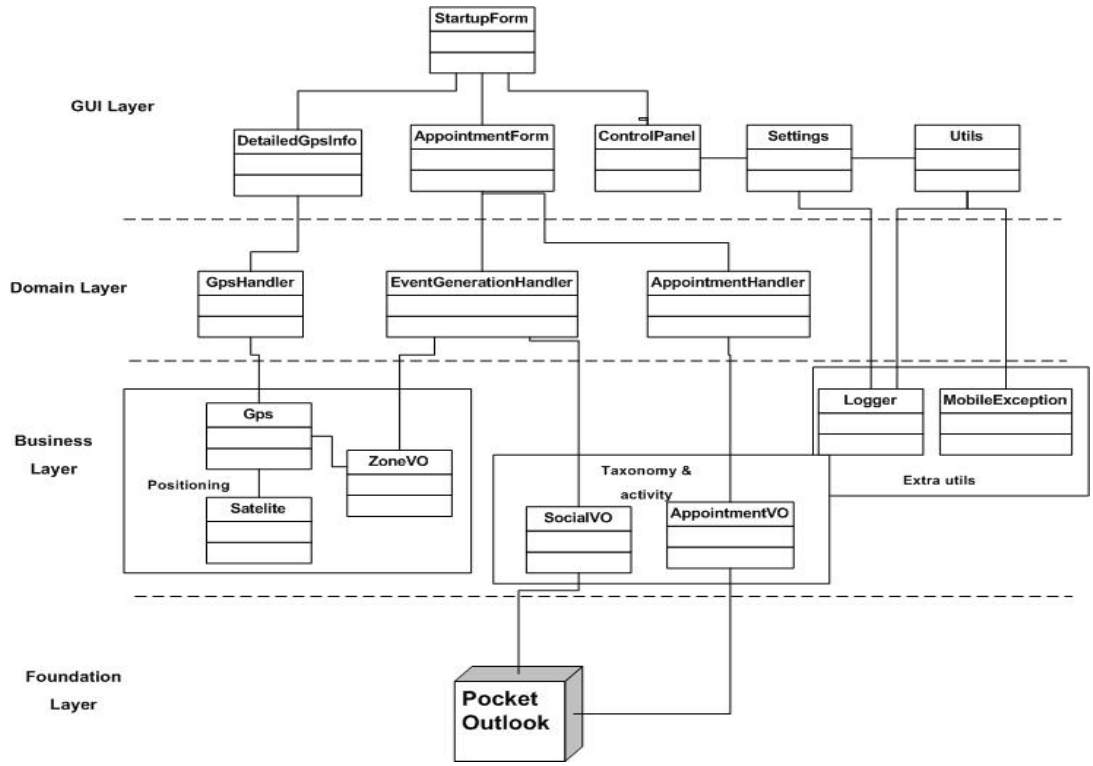


Figure 4-2: Application architecture

The application is built following Larmans' layer principle (2002), clearly separating the architecture into individual layers with their respective responsibilities and services. The figure shows the separation into the four main layers, with the business layer as the most important one. This is where the computation of the user context is conducted and where the main business logic is implemented. The context calculations are conducted at two separate occasions, one being when the user has moved (more than 40 meters) and the other when there is a change in activity. The result from the user-context calculation is the determining factor for whether or not this information is presented to the user. Next I will give a description of the three modules upon which context-aware computation is based.

4.4.2 Activity Module

This module stores all user-entered activities (appointments and events) in a central storage on the device, in the Pocket Outlook database. The responsibilities for this module are to be able to communicate with this database and to be a handler of events generated in the Pocket Outlook database. Calendar communication is done through Pocket Outlook Object Model interface (POOM) (Microsoft, 2012a) to achieve contact with Microsoft Pocket Outlook storage. This interface enables the possibility to retrieve and manipulate all the information in this storage. The daily maintenance and updates to this storage is done through the Windows Mobile calendar interface.

4.4.3 Social Context Module

This module is responsible for storing and treating the user's social context. Since the apprehension of context will be individually based, a taxonomy is applied to group the different options available. The taxonomy, available below (Table 4-1), is carefully chosen with terms and concepts interpreted in common by most people.

Table 4-1: Social Context Taxonomy

Categories	Leisure	Work	Travel
Sub categories	Shopping	Meeting	Train
	Cinema	Preparation	Car
	Spare time	Own time	Tube
	Food	Travelling	Foot
	Culture	Phone meeting	Transport

Building on the work of Prekop and Burnett (2003) the social context is applied as tags to the Pocket Outlook activities and stored as meta-information together with the appointment. The user interface implementation of this is programmed into the Pocket Outlook object model as an extension to the Pocket Outlook category interface, creating a seamless experience for the user. Users can then enrich (meta-tag) their activities and events through a familiar interface (Figure 4-3).

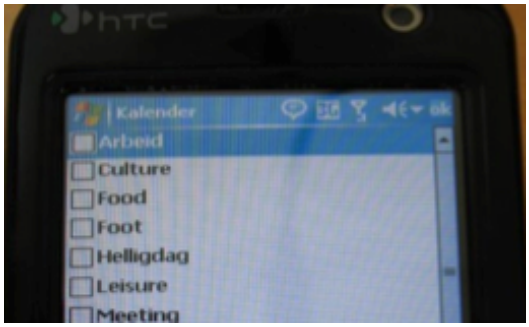


Figure 4-3: Pocket Outlook category interface

4.4.4 Location Module

The responsibility of this module is to calculate and store the most recent update of the user's geographical location. For exact positioning, an external GPS unit was used and configured through COM ports for a valid connection. Upon receiving GPS data, the string received is parsed and actual longitude and latitude coordinates are extracted. To be able to apply this information efficiently and to provide an extra layer of abstraction, the user's location was mapped to different city zones (Figure 4-4) and the user could monitor location through an admin panel (Figure 4-4).

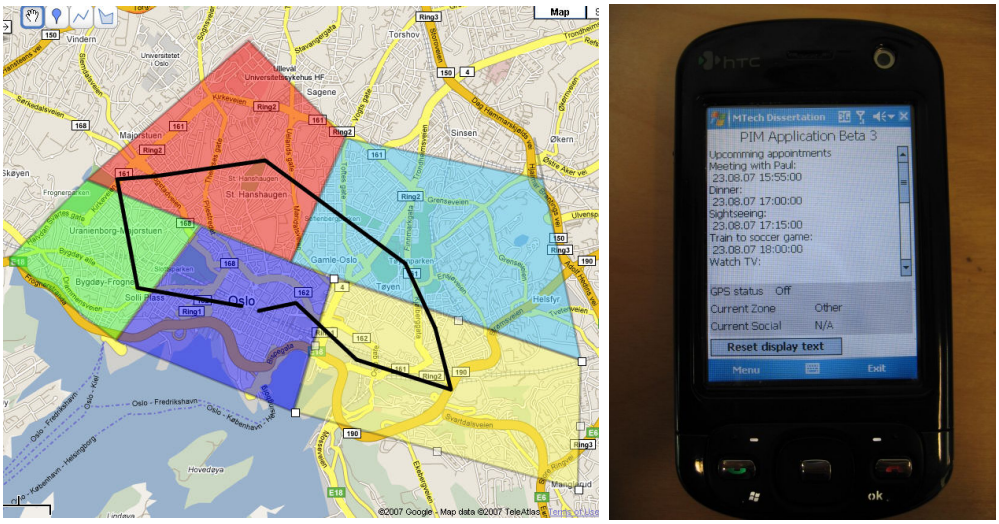


Figure 4-4 Left: City Map Right: Device admin panel with GPS logging

As the user is moving about, the actual location is updated and matched with the appropriate city zone and then stored locally. The separation into zones is based on the official city regions of Oslo (Oslo Kommune, 2004); however, in order to ease the visual presentation for the user only five out of a total of 16 zones are displayed in the map.

4.4.5 Context-Aware Computation

The underlying computation of the user-context is the important foundation of the application. This core module of the application coordinates the presentation of information to the user. An example of rules for calculation of context-aware information is: at system start-up, or when user position is altered with 40 meters or if a system event is triggered by change in activity. A code example for communicating with the GPS module is highlighted below:

```
//Registration of two events
      gps.DeviceStateChanged += new
GpsLib.DeviceStateChangedEventHandler(gps_DeviceStateChanged);
      gps.LocationChanged += new
GpsLib.LocationChangedEventHandler(gps_LocationChanged);
//Calling the event update method
// update the UI on the UI thread
Invoke(updateDataHandler);
```

This enables precise tailoring and adjustment of information for each user and thereby a presentation of only relevant information. The sequence of calculation is described in Figure 4-5.

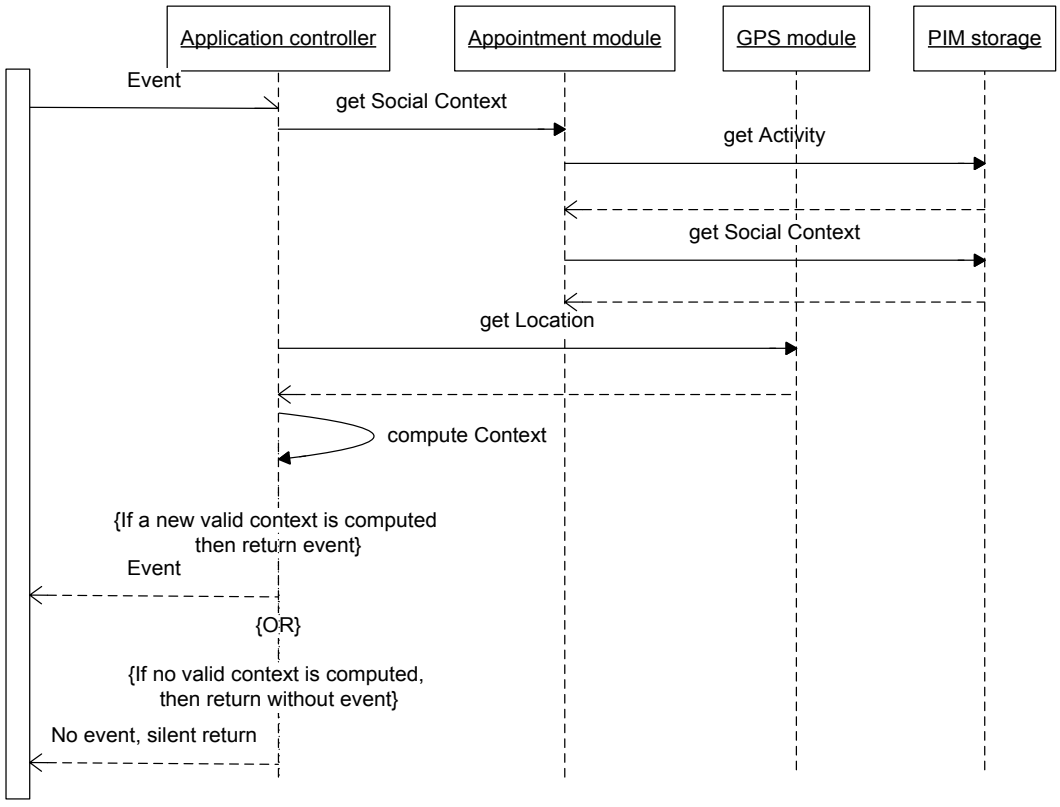


Figure 4-5: Sequence diagram of context computation

Nonetheless, there might be times when disturbance is not appropriate, for instance if one is attending a meeting, being at the library or whilst driving one's car. On such occasions one's social action (being at the library) or activity (driving a car) can be the weighting factor forcing the application to suppress an otherwise legitimate information event. The application is able to deal with this, and will present the user with missed events once his/her user-context changes to a non-intrusive mode. Below is an example of such a possible message shown (Figure 4-6).

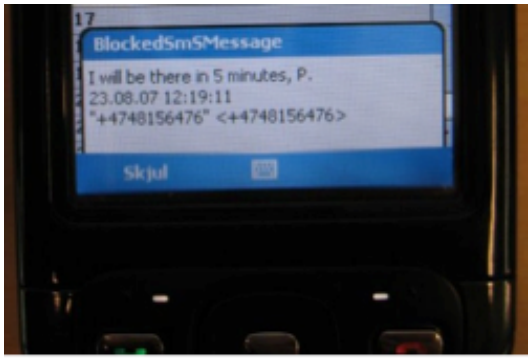


Figure 4-6: Missed event, displaying the blocked SMS

The function for determining the context-aware setting into the user-context is, as earlier described, an important underpinning for the application. To be able to describe the actual composition of the functions, the following variables are used: α describes social setting, β describes location and γ describes current activity.

The function f of context can then be described as follows: Context = f (α, β, γ). Based on this function we can then apply rules for each situation and thereby have the system act differently in each unique situation. In a given situation context can be expressed as the combination of social setting, location and activity:

$$Context = f(\alpha, \beta, \gamma)$$

In this function, social setting α can be represented as an element from the social context taxonomy matrix, given in Table 4-1. The details of α can be given as follows:

$$\alpha \in [s_{i,j} \mid i=1\text{-}3, j=1\text{-}5]$$

α is represented from the elements of the context activity matrix where $s_{i,j}$ is a combination of one element from 'Categories' with an element from 'Sub categories'. For instance, $s_{2,4}$ would represent travelling on work related activity.

The location, given by β, is decided by the users current GPS determined position. In the details for β given below, p_i is one of the city zones (please refer to section 4.4.4 for city zones information).

$$\beta \in [p_i \mid i=1\text{-}16]$$

The users activity, γ, is represented from an activity / appointment extracted from the device internal Pocket Outlook application. γ is any number from 0 up to n current activities (please refer to section 4.4.2 for details on the activity module). The details for γ is expressed below:

$$\gamma \in [a_i \mid i=0\text{-}n]s$$

To further detail and explain the implementation of the context-aware computation, the following code samples are included. The function for determining the context-aware setting into the user-context will be run at several occasions. One major occasion is in the case the user has moved a predetermined distance, i.e. 40 meters. Given below (Figure 4-7) is an excerpt showing the initialization of the function where a GPS event is received, and the chain of command for the context-aware computation is started.

```
if (!Opened) { //create handles
  newLocationHandle = CreateEvent (IntPtr.Zero, 0, 0, null);
  deviceStateChangedHandle = CreateEvent (IntPtr.Zero, 0, 0, null);
  stopHandle = CreateEvent (IntPtr.Zero, 0, 0, null);
  gpsHandle = GPSOpenDevice (newLocationHandle,deviceStateChangedHandle, null, 0);

  if (locationChanged != null || deviceStateChanged != null) {
    CreateGpsEventThread ();
  }
}
```

Figure 4-7: Code excerpt one from function for context-aware computation

From the following code excerpt (Figure 4-8), a snippet from the implementation of the context-aware computation is shown. This code snippet is focused from one part of the program where user action is decided. For a given user, with presence in a given city zone, action can be taken on the background of the current social status and on-going activity. In this proof of concept solution, the action taken towards the user were presentation of a website matching the computed user context.

```
else if (gpsLocation == 2) { //Zone 2, Centre, getCalculated
    if (social0 == "work" || social1 == "work") {
      if (social0=="meeting" || social1=="meeting") {
         createURLNotification("url", appVO);
      }
      else if (social0=="preperation" || social1=="preperation") {
          createURLNotification("http://www.regjeringen.no/nb.html?id=4", appVO);
      }
      else if (social0=="own time" || social1=="own time") {
          createURLNotification("url", appVO);
      }
     }
     if (social0 == "travel" || social1 == "travel") {
       if (social0 == "train" || social1 == "train") {
           createURLNotification("http://www.nsb.no/internet/jp/trafficdelays/index.jhtml", appVO);
       }
       else if (social0=="tube" || social1=="tube") {
           createURLNotification("url", appVO);
       }
       else if (social0=="car" || social1=="car") {
           createURLNotification("url", appVO);
       }
       else if (social0=="foot" || social1=="foot") {
           createURLNotification("url", appVO);
       }
     }
     if (social0 == "leisure" || social1 == "leisure") {
       if (social0=="shopping" || social1=="shopping") {
           createURLNotification("http://www.oslocity.no/site/page.aspx?Page=Forside", appVO);
       }
       else if (social0=="cinema" || social1=="cinema")
       {
           createURLNotification("url", appVO);
       }
     }
}
```

Figure 4-8: Code excerpt two from function for context-aware computation

4.5 Participants

A total of 15 persons, 2 females and 13 males aged between 20 and 40, participated in a real-world evaluation. The test persons varied in gender, age and technological experience according to the following table.

Table 4-2 User breakdown in sex, age and technological expertise

	Very experienced with technology	Medium experienced with technology	Little experienced with technology
Male, age 20-25	4	1	
Female, age 20-25			1
Male, age 25-30	4	2	1
Female, age 25-30		1	
Male, 30-40		1	

4.6 Material

The application is designed for and implemented on a HTC 3600 Pocket PC phone running Windows Mobile 5. This device was chosen as it, at the time, had a powerful hardware specification and a touch screen interface for manipulation (for a full specification, please refer to the Material section of chapter 3). At the time of testing, the participant was given the phone device, as well as a separate sheet of instructions for how to operate it (Available in Appendix B)

4.7 Procedure

The user testing was accomplished by having 15 persons individually use the application for approximately half a day each (around 4 hours). The test consisted of the persons using the developed prototype application on the Windows Mobile phone that was handed out to them. Whilst using the device they performed practical tasks as given in the task list. At the beginning of each test session, the candidates were given this task list (Table 4-3) and s/he had to enter all appointments into the application. This ensured all participants having the same application set-up.

Table 4-3: Application set-up information entered by the users into the device

Appointment Subject	Location / Zone	Start	Categories	Comments
Train at central station	2	12:00	Travel, train	
Shopping at Oslo City	2	+ 20' min	Leisure, shopping	
Meeting at company X	1	+ 20' min	Work, meeting	Sensitivity: private
Sightseeing in Old Town	1	+ 20' min	Travel, foot	
Tour at Munch Museum	4	+ 20' min	Leisure, culture	
Coffee break at St. Haugen	5	+ 20' min	Leisure, food	
Shopping in Bogstadveien	5	+ 20' min	Leisure, shopping	
Travel from Frognerparken to Solli plass	3	+ 20' min	Travel, transport	
DNB Nor Solli plass	3	+ 20' min	Work, meeting	Sensitivity: private
Government	2	+ 20' min	Work, preparation	
Central train station	2	+ 20' min	<None>	Route stop

When performing the test, all participants moved through the city according to the scheduled events. During the evaluation, the user-context was continuously monitored and calculated, and where action appropriate accordingly to the rules were taken by the application. When position and activity would match, the application also checked the social category before taking appropriate action. If the user were in a context setting available for receiving information, the device would present information. On the contrary, if the user was in a busy mode, or otherwise not to be disturbed, the device would cache all events and incoming SMSs thereby leaving the

user alone. Once the user returned to normal mode, a summary of missed events would be displayed. By the end of the evaluation, each user was given a questionnaire (please refer to Appendix A) to complete. These questions covered three different aspects of the prototype: "Initial use", "PIM in action" and "Overall statements". The results from these questionnaires are presented and evaluated in the next section.

4.8 Results

This research focuses on the implementation of a context-aware mobile application and its ability to tailor context-aware information depending on the calculated context. The questionnaire designed for this experiment, which is given in full detail in Appendix A, reflects this fact and comprises statements related to the participants' experiences when using this application and receiving tailored, context-aware information. Users' opinions are expressed on a six-point Likert scale ranging from: *strongly disagree, disagree, mildly disagree, mildly agree, agree*, to *strongly agree*. Each option was coded with number from 1 to 6 respectively, and thereafter analysed by calculating the mean and running a T-test. An extraction of the results from this evaluation is presented in the forthcoming table.

Table 4-4: Results from user evaluation

	Statement	Mean	t-test
1	The information provided by the reminder system correctly matched my current location	4.80	**0.000**
2	The information provided when I was "Sightseeing in old town" was incorrect.	2.47	0.310
3	The summary of blocked events I received after appointment "DNB Nor Solli plass" was useful	3.73	0.135
4	The system provides duplicated information	2.73	0.524
5	I liked the fact that the application is integrated with Outlook	5.47	**0.000**
6	The reminder system is useful	5.20	**0.000**
7	I would use this application in my daily life	3.93	**0.005**

Table 4-4, and from Figure 4-9, one sees that all answers display a positive bias (Statements 2 and 4 are in fact negative statements about the application, therefore the negative bias here reflects positive user statements). The questions with a pattern of statistically significant responses have been boldfaced. In question 1, 14 out of 15 participants answered that the information displayed did match their current location and the responses are statistically significant. This indicates a correct calculated user-context. For statement number two, 11 out of 15 respondents were negative to the fact that the information being displayed in one appointment was incorrect. This indicates that 11 were shown correct information and four had incorrect or no information displayed. There is thus a strong bias towards negative answers, a few (4) positive answers and no middle values (MA / MD). This polarization of results however leads to the data for this statement not to be statistically significant.

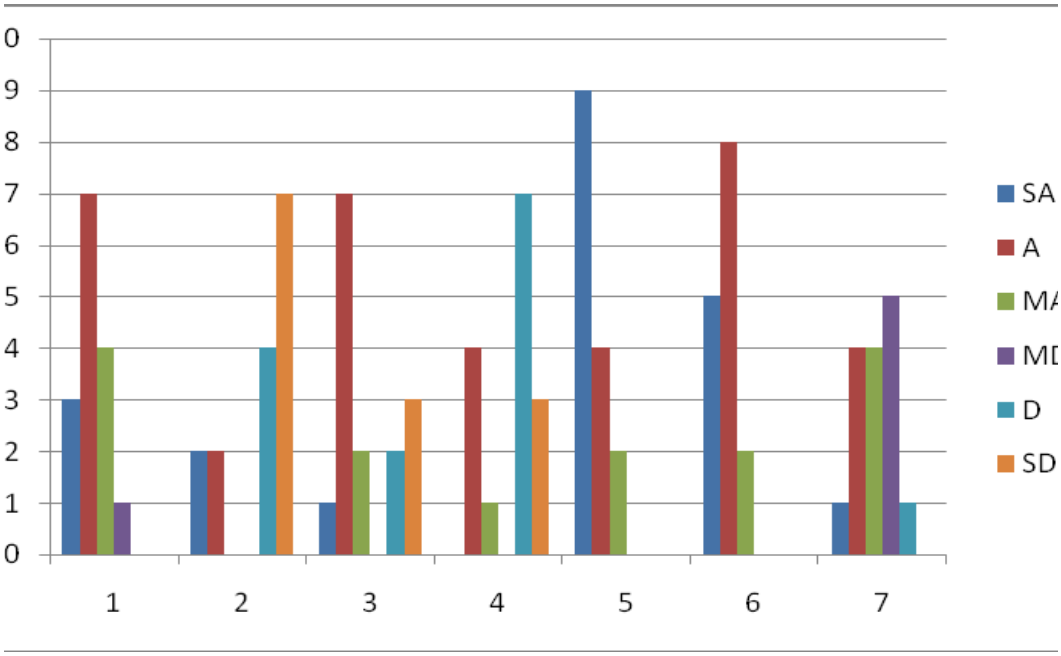


Figure 4-9: Graph displaying user evaluation results

For statement number three the majority of respondents point towards this being a useful feature, which makes this an interesting feature to pursue further. This also strongly indicates that a successful user-context computation was performed, and that system generated events received whilst in a non-disturbance mode can be successfully presented to the user when again available. The unanimous response to question five points strongly towards the appropriateness of extending a known

interface, Pocket Outlook, with extra meta-tags for enriching the appointment information. In statements six and seven the user's interest in such applications for use in daily life and assessment of usefulness is further explored. The respondents indicate that although they find the application useful, it is not necessarily something they will be using daily in their lives.

4.9 Discussion

The application has shown the feasibility and possibilities for tailoring information specific to user's needs. The context categories, given by the taxonomy, will need adaptation if such an application were to be pursued further since the application currently does not have the ability to learn from the users' actions. A possibility to adapt the taxonomy to the given domain, or for the application to learn from user action could be a feasible solution. This is in accordance with the findings of Prekop and Burnett (2003) which state that it is very difficult to determine environmental information sources. Almost every source of information would need to be tailored or have a custom interface written in order to be able to deliver its services to the application. When these issues are taken care of, the user experience might improve and increase even more the positive trend in answers to question six and seven. Moreover, as shown by Zhou et al. (2006), information tailoring is an important task to help users interpret data. This application focuses on tailoring by having minimal information displayed at the same time, when new messages are shown, thereby easing the users' interpretation.

Currently, calendars and applications based efforts do not to a large enough extent take multi contextual information into account (Häkkilä and Mäntyjärvi, 2004; Schmidt et al., 2001) as they often only reproduce the information already available there. Thereby, this can at worst lead to incorrect display of data and at the best a reproduction of data in a new interface. This developed PIM application greatly differentiates from this by only displaying information based on the computed user context, given by the three factors social context, location and activity / task. Earlier approaches that have made use of calendar data from Pocket Outlook, often end in using Pocket Outlook data together with a simple timeline (e.g. (Dey, 2001)). In our approach, the use of Pocket Outlook data is extended to not only retrieve and display the data, but also to add extra meta-information to the appointments. Results from

question five show that all respondents stated they liked the integration of the developed PIM application with the Outlook calendar.

In the evaluation exercise, generation of information was tightly connected with the actual task at hand and participants were asked to judge whether or not they found the application useful (question six). Results show that all 15 users involved in the evaluation thought the application gave useful value. This would indicate that the reminder system is an application with practical value for the users. As covered in the final question, the users were asked to state whether or not they would like to use this application in their daily life and one should note a separation in two camps, one positive and one negative. The positive responses highlight a possible opportunity and one should consider pursuing this in further research.

4.10 Conclusions

Context and context-awareness have long been acknowledged as important and have generated considerable research effort. Returning to the evaluation guidelines by Hevner et al. (2004), an assessment of research output from this experiment is given in the next table.

Table 4-5 Research experiment mapped to evaluation guidelines

Design Research Guidelines	**Completion**
Design as an Artefact	Considered complete based on the use of models to describe the architecture and with the developed prototype as artefacts
Problem Relevance	Considered complete through the identification of the research gap after reviewing existing work in literature review section and through the mapping between research objective and experiment
Design Evaluation	Considered complete by having the artefacts (models) evaluated in practice by code implementation and software testing. The final evaluation by users ensures validity and provides a ground for limited generalization
Research Contributions	As shown through the findings and the evaluation new knowledge about experiences from three-dimensional context-aware tailoring can be extracted from this work.
Research Rigour	Considered complete since SCRUM has been the underlying methodology for the software development and software craftsmanship techniques as well as eXtreme programming practices have been applied for quality assurance.
Design as a Search Process	Considered complete because the Design Research process model by Vaishnavi and Kuechler (2007) has been followed as a framework in this experiment
Communication of Research	Considered complete through the findings summarized in this conclusion and by building upon these results in experiment two (Chapter 5)

In this chapter the design, implementation and evaluation of an application prototype, which integrates context / context-awareness into a PIM from a novel three-dimensional perspective has been presented. Social-, geographical- and activity information was the chosen dimensions for context-aware information and integrated in a PIM allocation using Windows Mobile with the .Net framework. This realizes the objective: to *implement and integrate multidimensional context-aware functionality in a solution for a mobile personal information manager.* User evaluation of the proof of concept displayed a strong positive bias, highlighting its potential usefulness and applicability. The solution and evaluation show the viability and usefulness of such an approach and I do believe that tailoring information in the manner described in this research takes the PIM concept one step further towards the ideal of providing tailored and timely information to mobile information users everywhere. For further research, exploitation of context-aware information in different scenarios for context taxonomy adaptation would be interesting, as would implementation of more location aware information to enhance the user experience.

Chapter 5

The Context-Aware Meeting Room

5.1 Introduction

This chapter explores people's daily activity when attending meetings. Not only will one have to attend a meeting, but one is also required to keep track of the daily schedule, meeting agendas, write meeting minutes and more. Applications like Microsoft Outlook or Lotus Notes can help with some of the tasks like keeping track of schedule and administrating meeting invitations and responses. Still when it comes to meeting minutes and direct involvement in the presentation the tools often come short. As identified in experiment one in chapter four, interesting opportunities for further research were the use of more location-aware information to create richer context-awareness.

This chapter introduces an intelligent meeting room scenario and answers objective two as stated in the literature review: to *explore the integration of context-aware information from cloud-based services and applications in heterogeneous mobile applications.* The developed system integrates the notion of location-awareness and investigates a heterogeneous mobile context-aware platform in relation to a cloud-integrated solution. To be able to answer the research objective, a series of tasks needs to be accomplished:

- Define sources of context-aware information
- Define the heterogeneous mobile environment
- Decide upon cloud and client communication architecture
- Implement a cross-platform messaging format
- Perform user evaluation to measure system viability

5.2 Literature Synopsis

Ahmed et al. (2005) presented in their work a *Smart Meeting Room*. Their contribution investigated how participant support and user clusters could be facilitated in an intelligent meeting room scenario. The concepts of user clusters are interesting and they highlighted how users from different clusters could be found ad hoc groups and collaborate. Effortless meeting scheduling, room allocation and updating related documents during the meeting and providing a communication facility were demonstrated. Information distribution was also amongst the goals of their research project and they showed how user groups could receive and collaborate based on information within a secure and reliable setting. Context and situation information were aggregated using location techniques. Importance in the solution was focused on providing these services as a separate higher layer, free from low-level implementation challenges. Other low level functions such as automatically muting the phone, authorization checks on documents were also hidden and handled automatically for the user (Ahmed et al., 2005).

There are several other research contributions that focus on the meeting room scenario and some also in relation to context-awareness. Research from Chen et al. (2004) proposed the use of Bluetooth as a key concept and technology in a smart meeting room. Their research concerns how to provide relevant services and information to the meeting participants based on their context. These services included presentation-, lighting control- and greeting service. Bluetooth was the component used for registering participants and becoming aware of their presence in the meeting room.

Cloud computing focuses on sharing data and makes it possible to distribute storage and computations over a scalable network of nodes, with large IT companies like Microsoft, Google and IBM, all having initiatives relating to cloud computing (Mei et al., 2008). One of the key features under this paradigm is scalability on demand, where the user pays for the amount of computation and storage that is actually used. Moreover, this scalability is usually completely transparent to the user. Mei et al. (2008) have focused on finding interesting topics for future research in the area of cloud computing, and have looked into how one can achieve similarities and

cooperation between the fields of pervasive computing and cloud computing services. Based on this, they identify some important areas for further research including: pluggable entities, data access transparency, adaptive behaviour and automatically assessment of application quality. When iterating we can see this in relation to the needs defined from Ahmed (2005) such as the ability to provide services at higher abstraction layers and automated discovery, administration and distribution.

5.3 Experiment Scenario

Let's meet Susanna on the way to a meeting in the corridor at her workplace. She looks up the meeting invitation in the calendar application on her smartphone and quickly finds her right way. Approaching the meeting room a server running in the meeting room picks up her Bluetooth ID and recognizes her as a meeting participant. The server application prepares a welcome message to be delivered to her. Susanna enters the meeting room and finds her place together with the other delegates. She starts the intelligent meeting assistant installed on her smartphone and is welcomed by a message from the server. The presenter enters the room, welcomes the audience and starts his presentation by moving a slide forward. Our user's device wakes to life as the intelligent meeting assistant kicks to life. Since the presenter changed slide, the notes for the particular slide were sent to the user's device. Through the slide series presented, Susanna is for each slide pushed additional information supplementing the presenter's slides. While enjoying the presentation she is able to cycle back and forth through the received notes courtesy of the application's functionality. When the presentation is finished, she disconnects from the presenter and finds her recently received information neatly organized together with previous meeting minutes in the history section of her intelligent meeting assistant.

5.4 Application Design and Architecture

The system allows a presenter to hold a presentation for an audience, both equipped with smartphone devices. During the presentation, slide notes accompanying the presented slides are pushed from the Google cloud as additional information to all participants' smartphone devices. The presenter device is used as a remote control

for the presentation. Google App Engine cloud servers host a server application, keeping track of meeting scheduling and meeting participants. A conceptual overview is presented below (Figure 5-1).

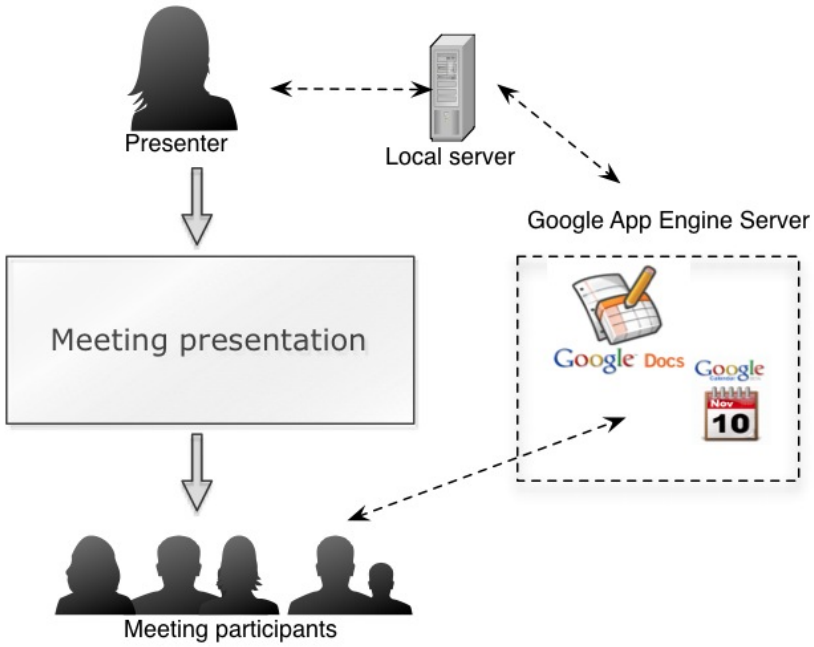


Figure 5-1: Conceptual overview of solution

The developed system consists of three major components:

- An Android based *presenter application*
- iPhone, Android, JavaME and Windows Mobile based *client applications*
- Java based *server applications*

The overall setup consists of the different mobile phone applications' connection to the server using wireless network (WLAN) and Bluetooth. The local server application uses proximity technology for detecting nearby clients and detects whether or not this is a registered meeting participant. When a meeting has started the presenter controls the application, and when s/he advances in the slide series, a small text-based meeting note accompanying the slides is sent to all participants.

Each of the three parts represents major parts of the application-suite. The role of the presenter application is to be used by the presenter as a remote control for the presentation. This application is equipped with functionality for starting and ending the presentation as well as moving back and forth. This is accomplished by the

Google Cloud integration with the Google Docs. When the presenter starts the presentation, the relevant slide series is identified in Google Docs and brought up. When the presenter is advancing, a message is sent to the local server and an event is triggered. The event handler notifies the cloud-based Google server about recent action, and this cloud server is responsible for advancing the slides, as well as finding the note associated with the current slide and distributes it by push technology to the registered participants.

The client application, namely the intelligent meeting assistant, has two main roles. The first is to assist the meeting delegate during the presentation by receiving meeting notes sent by the presenter and to provide functions for moving back and forth in the meeting notes. The second role is to act as storage for meeting history. The user can go back to meeting notes from previously attended meetings and browse back and forth in these. The client application needs to be manually started, but will then be automatically recognized by the local server. Context-aware information in terms of proximity and user identification is applied in order both to identify a registered participant and to communicate with it.

The server application acts as hub and an agent between the client applications, of the meeting delegates, and the presenter. This functionality is enabled by implementing a Java server application on a local machine, and by having a Google cloud based server. The local server instance has client discovery and meeting administration as its main tasks. Client discovery is performed through Bluetooth proximity identification. Clients successfully identified and acknowledged as meeting participants are added to a list in the local server. The interface (Figure 5-2) of the local server will keep track of current and forthcoming meetings, together with this list of recognized participants.

Figure 5-2: Local server interface

Polling the cloud server provides the participant list and the cloud server identifies this based on database lookups and by harvesting calendar information. This means that all business logic is deployed to the Google cloud which allows for scaling in times of heavy traffic / load, thus avoiding jamming the local server. The cloud based server used integration with external features such as Google Calendar and Google Docs. All information sent between the clients and the servers and between the local server and the cloud server used a standardized message format. This enables efficient exploitation of bandwidth and created the opportunity for cross language and cross platform sharing of information. The sharing of information is made possible by the format of the protocol buffer messages and their predefined message structure. The Google cloud services were also used for entering appointments, inviting participants and keeping track of the upcoming schedule.

By combining context-aware computing and cloud computing strengths of the two worlds are combined. Cloud solutions in form of Software as a Service (SaaS) such as Google Calendar and Google Docs contributes two richer application content. Presentations from Google Docs can be enriched with full feature functionality as available from a desktop computer and in the Google Calendar the appointments are edited through a familiar interface facilitating for meeting scheduling and easier input of meeting details. This expands possibilities from the mobile clients by leveraging cloud-computing functionality. Following research by Guan et al. (2011) this is in line with one of their proposed architectures for mobile cloud computing,

the agent-client scheme. This is an elastic approach and the security is addressed by build in security measure of the Google cloud services. Additionally the developed prototype system benefits from the Google Platform as a Service (PaaS) by having the application deployed to the Google App Engine. This effort builds on offloading of tasks (Guan et al., 2011; Simoens et al., 2011) easing the burden on the mobile clients and thereby prolonging battery life and preserving resources.

5.4.1 Application Architecture

The architectural backbone of this system suite is the flow of messages. When incorporating four different mobile platforms, a cloud based server and a local server the heterogeneity is high; equally high is the complexity of message formats and standards. An early architectural decision made was to use a common network protocol for all application platforms, including the server and client applications. From earlier conducted implementations and from my own programming experience it was clear that none of the different application platforms had any message or data type in common. Therefore I looked beyond the platform and landed on the Protocol buffer protocol (Google, 2011). Protocol buffers are contract first network protocol supporting a variety of modern programming languages including Java, iOS, Windows Mobile and Android (Google, 2011). For JavaME, a new protocol buffer implementation was created (Google, 2011) since there were no implementations supporting this platform. Protocol buffer messaging is reported as a considerable improvement in comparison with Web Service performance wise (Google, n.d.-a).

Figure 5-3 offers an overview of the technical architecture. The Google cloud server is the central unit in this system, containing all information and is responsible for managing the clients.

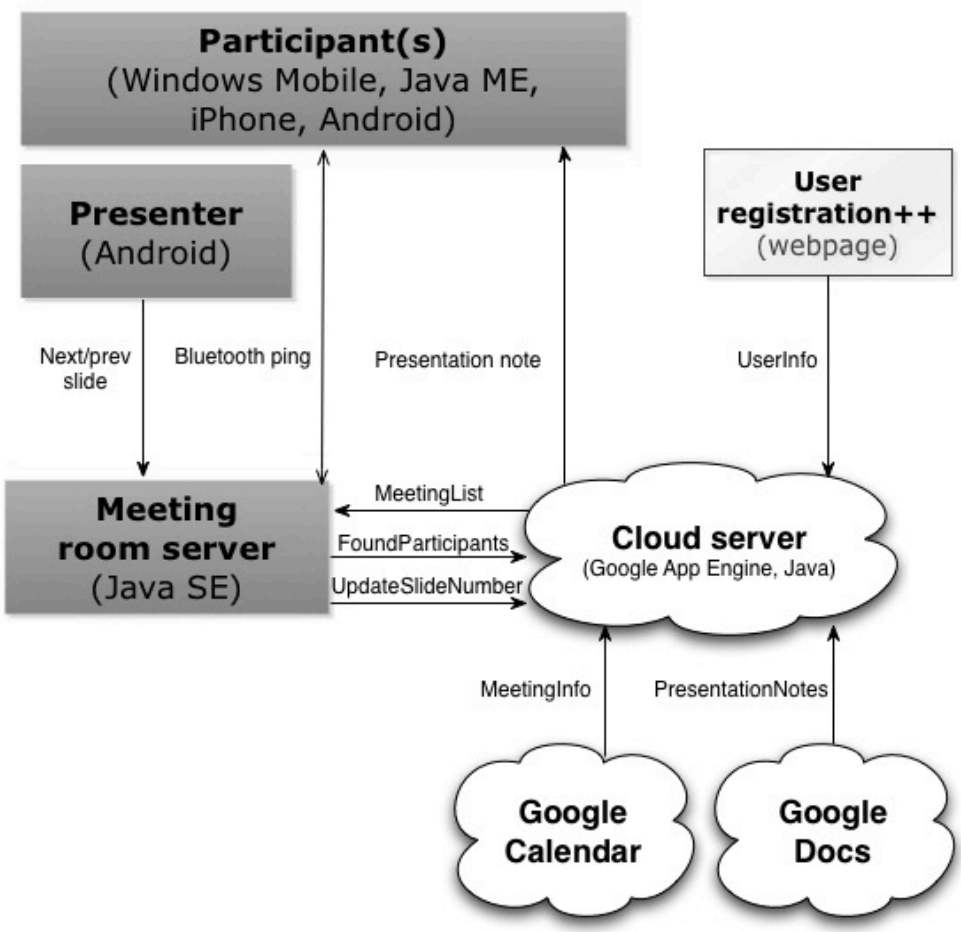


Figure 5-3: An overview of the system architecture

For new users, they will have to start by registering their device in the cloud server. This is done in order to associate the device Bluetooth ID with a user profile. The meeting room server and the cloud server will communicate over HTTP. Information including list of meeting participants and updates in slide numbers are exchanged. When the clients are discovered and connected to the meeting room server, they receive a valid server address and perform a HTTP Get request to this address in order to complete the registration. The response from the server is in the form of a welcome message, protocol buffer format. During the meeting the presentation notes pushed to the clients from the cloud server are protocol buffer implemented, meaning all clients get pushed the identical message.

5.4.1.1 Presenter Application Module

The presenter application is the simplest application of the system. The purpose of this application is to act as a remote control of the slide show for the presenter. The user interface consists of a button for next and previous slide and a large text-area for

status information (Figure 5-4). When the presenter presses one of the buttons a HTTP POST message is sent to the local server and transformed into a similar named event triggering action. The presenter application was implemented for Android devices based on Android 1.5 SDK.

Figure 5-4: Presenter application

5.4.1.2 Client Application Module

A heterogeneous client environment was developed consisting of four different client platforms (Figure 5-5). The functionality of the clients concentrated around two features:

1. Presentation assistance: Receiving meeting notes, as the presenter was moving forward in the slides. In these received notes it is possible to browse back and forth in those received.
2. Meeting-minute history: All slide notes received in each meeting are archived locally in the device organized by meeting. This enables later offline reading of previous meetings.

WLAN is the communication protocol implemented for client server communication. All meeting notes received are transferred using the described protocol buffer format. This enables a neat cross platform approach with the same message valid for all devices. This also enables client-to-client interaction, if so desired. The clients are designed with simplicity in mind, keeping the number of available functions low. To achieve this goal the server in the Google cloud handled all heavy load and business logic.

Figure 5-5: Left: JavaME client. Right: iPhone client

5.4.1.3 Server Application Modules

The server application modules consist of two separate servers. One is the local desktop server and the other one is the Google cloud server. The reasons for using two server applications are twofold. Firstly, the local server is needed in order to discover and connect with the meeting participants that attend the meetings. Secondly, to ease integration with cloud services such as calendar and presentations, a cloud server was chosen. The other benefit of the cloud server is the scalability. A meeting room scenario can have many participants, and in such a case the cloud server will also handle scalability.

The local server was implemented on a Jetty instance, and was programmed as a multithreaded application due to its many concurrent tasks. The main task of the local server is to scan for Bluetooth devices continuously, while also keeping track of connected participants.

The cloud server application contains all business and data access logic. This server integrates with Google services such as the Calendar and Docs. All presentations

used are stored in Google Docs with the meeting notes attached as meta-information. A custom written parsing implementation of the Google Docs document was added to be able to harvest presentation data. The step of harvesting information from the notes from Google Docs is an important feature of the system architecture due to the fact that this loosely couples the presenter, the presentation and the machine from which the presentation is displayed. Registered participants, meeting information and scheduling, were all stored in the Google cloud. Using their API I was able to query for relevant information. To connect the Google Calendar with the correct presentation in Google Docs a system of context-aware XML tags to integrate the features was used.

5.4.1.4 Context-Awareness Features

Context-awareness features were implemented in respect of user proximity, user status and context-aware tagging of cloud based information. With regards to user proximity, Bluetooth was used for identification. Initially a classical Bluetooth approach for searching was used, but two major limitations came to surface: Most newer phones can only be discoverable for a limited period of time, and in cases with a high number of participants the usual Bluetooth dialogue would suffer a too large delay in order to connect all devices. This was solved by applying proximity technology and by constructing the feature *Bluetooth ping*. This approach is implemented by pinging the Bluetooth hands-free service at nearby devices. This service is always running and there is no need to be in discoverable mode. By doing so, it was feasible to employ the knowledge of the preregistered Bluetooth address and by ping it was possible to discover who was close to the meeting room and thereafter only connect with registered participants. This enables Bluetooth identification and connection without having to deal with unknown devices as well as avoiding doing a costly search operation.

5.5 Participants

The research experiment had two main phases of evaluation. In phase one, a pilot test was performed with a total of 12 users. In the second phase, the main evaluation, another 40 people participated. All 12 users from the pilot test and the 40 from the main test session were aged between 20 and 55 years old. All participants had

previous knowledge of mobile phones and mobile communication, but had not previously used the type of application employed in this experiment. None of the pilot test users participated in the main evaluation. From the user computer experience classification (asserted based on a questionnaire employing the taxonomy of McMurtrey (2001)) it was identified that the majority of the users had a good level of computer expertise.

5.6 Material

The material used in this test consisted of four different mobile phone devices (Table 5-1). They were chosen as each of them represents one mobile operating system and this fact supported the need for a heterogeneous environment.

Table 5-1: Test material mapping

Device	Operating System	Used by
HTC Magic	Android version 1.5	Presenter and participants
Nokia E61 S60 3rd edition	Symbian 9.1	Participants
Apple iPhone, 1st generation	iOS 3.1.3	Participants
HTC TYTN II	Microsoft Windows Mobile 6.1	Participants

For server hardware a MacBook Pro 2.4 GHz Intel core 2 Duo with 4 GB of RAM was used. The main tasks of this machine were to scan for meeting participants and handle registration as well as the messaging services. In addition to this material the Google App Engine cloud server was used for the backend server application.

5.7 Procedure

The main user evaluation was accomplished by having 40 persons attend different meetings and help test the application functionality. The context-aware meeting room prototype application was evaluated in several separate test sessions, with each session consisting of between five to eight participants.

In each test session the users started off by receiving a test-device and then they entered a given web address to register. Thereafter, the user left the room and entered the given meeting room, and would be welcomed with a message from the server. When all participants had arrived, a 30-minute presentation started and meeting notes from the 12 slides were pushed to the users devices (Figure 5-6).

Figure 5-6: User evaluation

The participants were asked to verify that meeting notes were received when the presenter changed slide. The users were asked to move back and forth in the received meeting notes before disconnecting the presenter. After the meeting ended, they were asked to use the history feature to view notes from a previously held meeting. To secure full offline functionality they were asked to move back and forth in the archived meeting note. The participants were given a list of the tasks (Please refer to Appendix D) to perform during and after the presentation to make sure everyone completed all steps. By the end of the evaluation, each user was given a questionnaire (Please refer to Appendix C) to complete. The results from these questions are presented and evaluated in the next section.

5.8 Results

The results presented here illustrate different parts of the questionnaire: statements one and two from the *user interface* category, statements three and four concern

performance, statements five and six are about *user features*, statements seven to ten from the category *context-aware information*, statements 11 to 14 from *application features* and statement 15 from *overall impression*. The statements are given below (Table 5-2) together with the mean, standard deviation and results of applying a one-sample t-test.

Table 5-2: User evaluation questionnaire and results

	Statement	Mean	Std. Dev.	t-test
User Interface:				
Statement 1	It is easy to see the available functions	3.58	.549	**.000**
Statement 2	The features of the application are hard to use	1.50	.555	**.000**
Performance:				
Statement 3	Browsing back and forward in the current meeting meeting-notes is quick	3.58	.549	**.000**
Statement 4	The presentation notes displayed on the mobile device was synchronised with the audio presentation/presenter's narrative	3.13	.686	**.000**
User Features:				
Statement 5	I was not able to register as a participant in the web application	1.25	.588	**.000**
Statement 6	I was able to look at next and previous notes during the presentation	3.78	.480	**.000**
Context-Aware Information:				
Statement 7	When the presenter started the meeting my mobile device was found and I began receiving meeting notes once in presentation mode	3.83	.385	**.000**
Statement 8	The close integration with Google calendar is an inconvenience, I am not able to use the system without changing my existing or creating a new e-mail account at Google	1.83	.903	**.000**
Statement 9	I do not like sharing my personal information	2.08	.694	**.000**

	(like my name and e-mail address) to a service that stores the information in the cloud.			
Statement 10	Storing meeting information in the Google cloud and combining this with personal information on the device is a useful feature.	3.38	.490	**.000**
Application Features:				
Statement 11	I find the ability to navigate notes during the presentation useful	3.55	.504	**.000**
Statement 12	I do not find the "meeting note" history functionality useful.	1.75	.630	**.000**
Statement 13	I do not find the presenter/participant model useful for a meeting environment	1.68	.616	**.000**
Statement 14	I would like to receive more content during the presentation (images, videos, files, web-pages etc.)	3.53	.640	**.000**
Overall Impression:				
Statement 15	I find the context-aware meeting room application useful	3.33	.572	**.000**

5.8.1 User Interface

Statements one and two deal with the user interface and are presented in Figure 5-7. These results reveal positive facts about the interface and statement one shows that a total of 39 out of 40 found it easy to find the functions in the client interface. Following in statement two, a total of 39 out of 40 respondents also found the features easy to use (negatively formulated question).

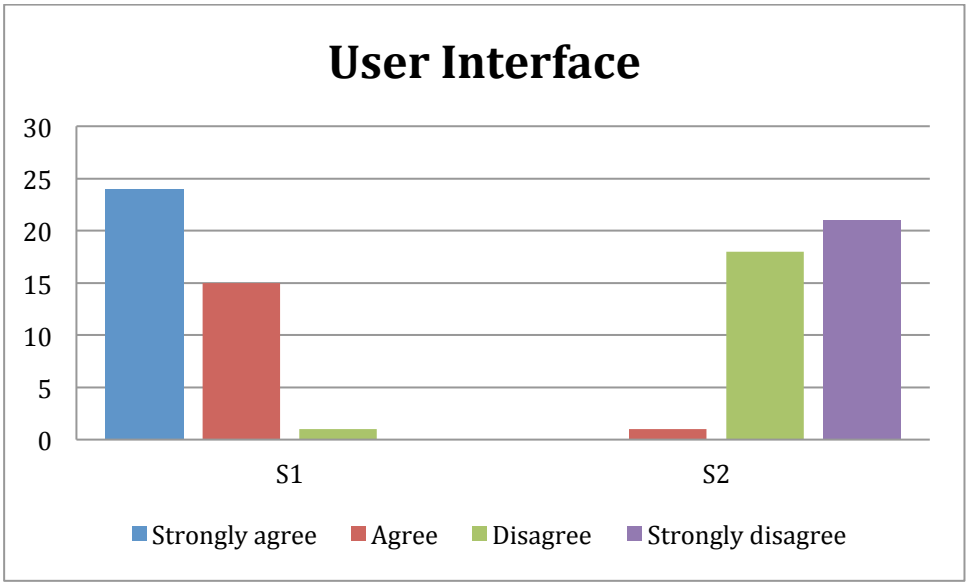


Figure 5-7: Statements regarding user interface

These results indicate that the user interface is no obstacle for using the application. The high score may be caused by the simplicity of the user interface, and it can be taken as an indicator of the interface not interfering with any tasks the user wished to perform.

5.8.2 Performance

The next graph (Figure 5-8) visualizes the answers from statements S3 and S4. These two statements deal with the performance of the application in order to verify correct behaviour. For both statements there is a strong positive bias, 39 out of 40 for statement S3 and 37 out of 40 for statement S4. This fact shows correct behaviour from the application and it fulfils the goal of cross-platform implementation. Statement S4 is also important in terms of detecting delays, as the participants will expect the device to be kept in sync and not tolerate any greater delay. From comments given to the test desires for additional functionality were discovered, such as a wish for better capability for organizing, searching in and sorting of the meeting notes after the end of a presentation.

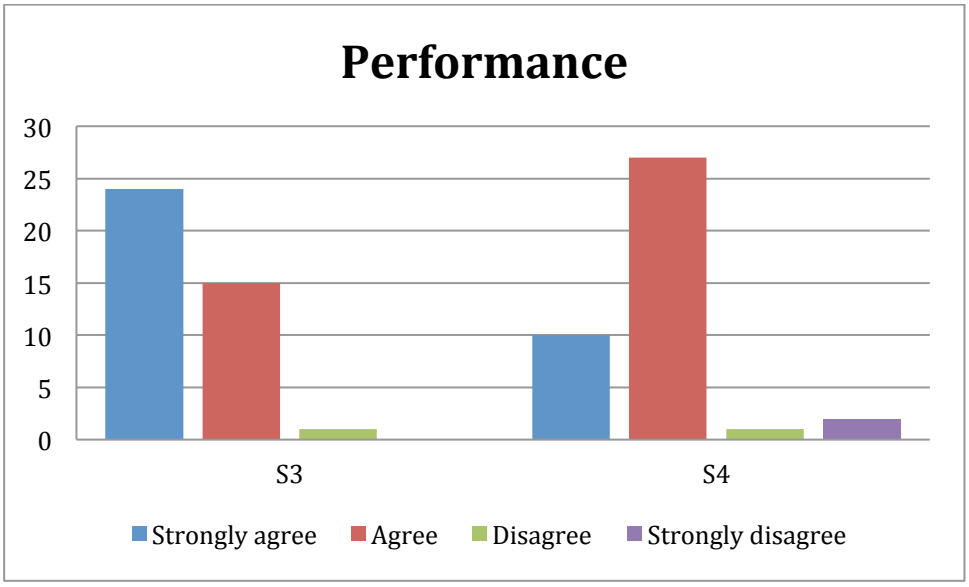


Figure 5-8: Statements regarding performance

5.8.3 User Features

When the participants were asked to take stand to the statement regarding *not* being able to register, the statistically significant results indicate desired application behaviour with 39 out of 40 rejecting the statement (Figure 5-9).

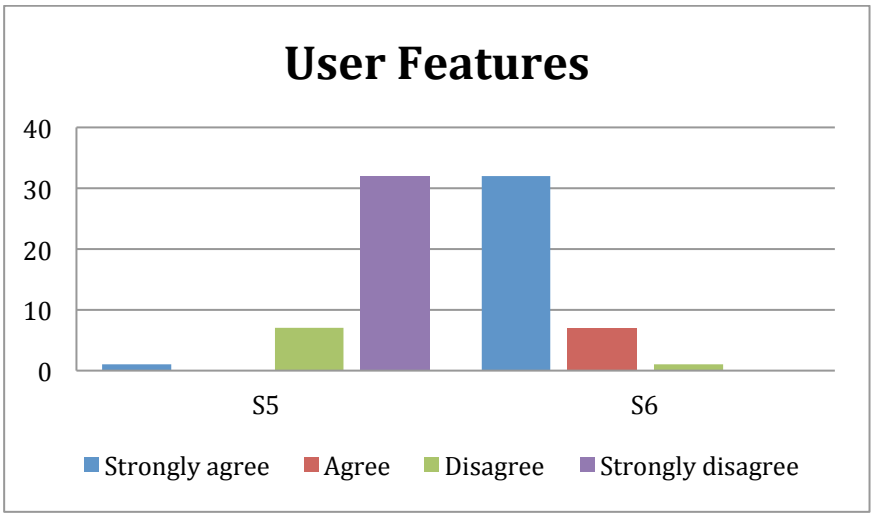


Figure 5-9: Statements regarding user features

The one not registered was taken care of by failover mechanism to be able to continue the test. Statement S6 received a strong positive response with 39 out of 40 respondents and a mean of 3.78, indicating proper behaviour of the functionality for browsing back and forth in received meeting notes during a presentation.

5.8.4 Context-Aware Information

In terms of context-aware information, participants were asked to take a stand in respect of four statements with results shown below (Figure 5-10). For the first statement (S7) a clear majority supported this assertion and indicated a correct behaviour from the cloud based application and correct cooperation with the local server. The next two questions (S8 and S9) cover the integration with the Google services. These results yield no clear response, but rather a mixed picture. Somewhat surprisingly, bearing in mind potential privacy concerns that could arise in this respect and despite the responses to statement seven and eight, almost all evaluators (both in the pilot evaluation and formal evaluation) agreed they are comfortable with the meeting information being stored remotely by Google in the Google service cloud (statement 10).

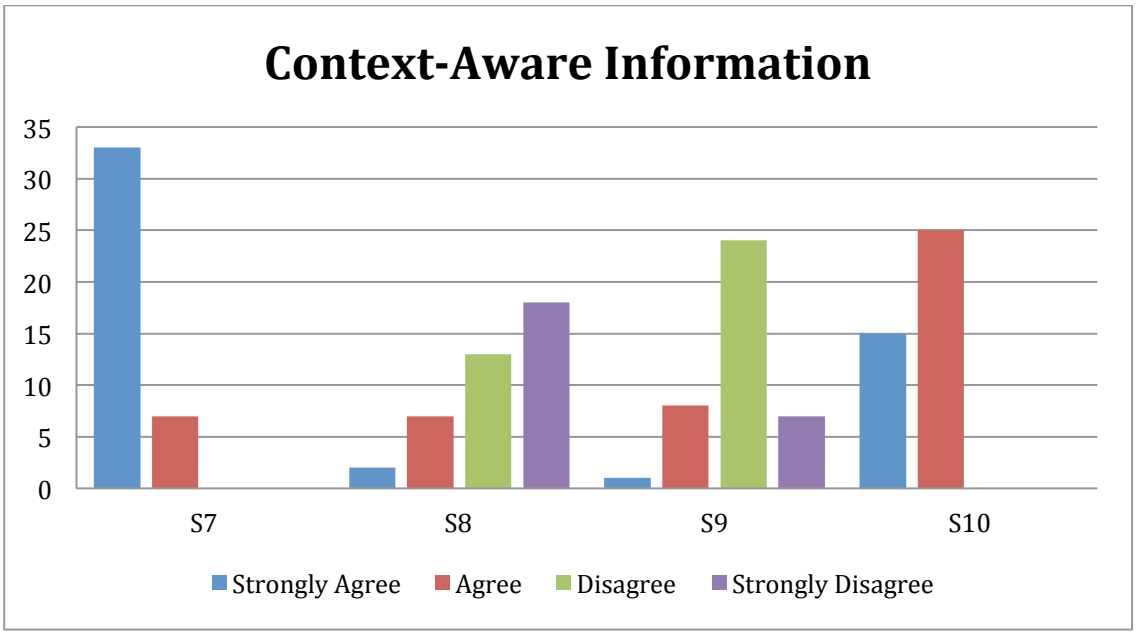


Figure 5-10: Context-Aware Information results

5.8.5 Application Features and Overall Impression

The participants were asked for an opinion regarding application features in statements S11 to S13. Results from these statements show unanimous responses and they are backed by the earlier statements dealing with expected behaviour of the application features (S3, S4 and S6).

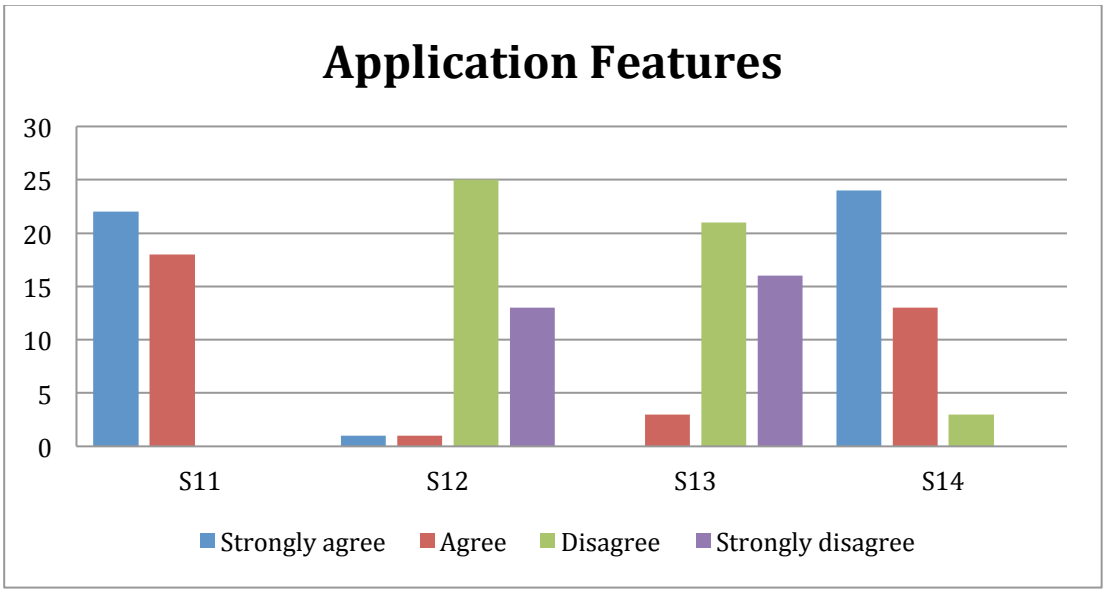


Figure 5-11: Statements regarding application features

In statement 14 the applicability of the application in a meeting room setting was tested. Statement 14 highlights the request for receiving more complex data such as images, video, animations and so forth. This is a reasonable point of view, and one could easily see this implemented to make the context-aware meeting room application even better prepared for real life scenarios.

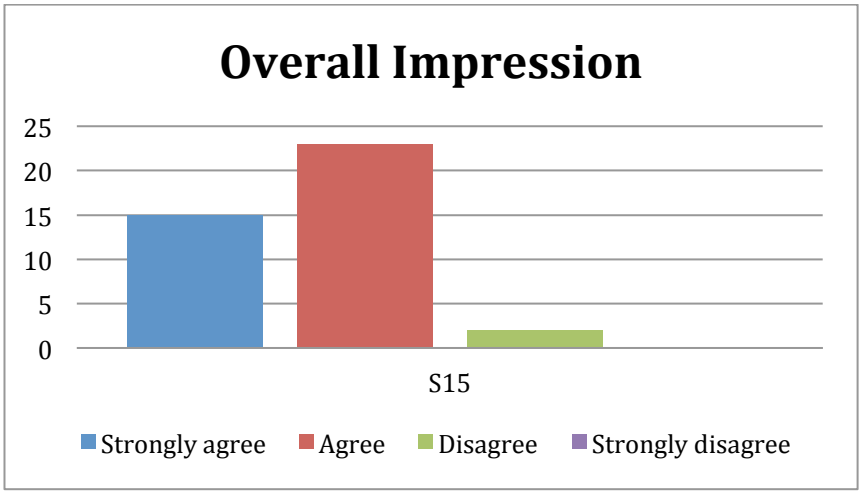


Figure 5-12: Statement regarding overall impression

The overall impression investigated in statement 15, shows the usefulness and received a very positive response from the users evaluating the application, as illustrated in statement 15 (Figure 5-12).

5.9 Discussion

I will in the following section elaborate, discuss and justify the research contribution of this experiment in relation to the literature.

Simplicity is the governing objective of client devices. A simple user interface eases the use of/interaction with the device (Lincoln et al., 2011). It is important to be able to have attention to the developed solution and the data it incorporates and presents. The results as presented from statement one and two support the user interface developed in this system. Both questions have an overwhelmingly positive bias. It is however important to keep in mind that the group of users evaluating the application were technically experienced. For this to be generalizable, more research needs to be conducted with different groups of users.

In respect of context-awareness, some interesting features emerged in this work. The ideas from Chen et al. (2004) where they propose the use of Bluetooth to create a smart meeting room are a foundation for this research. They present and pursue the concepts of using context information as a basis for providing participants with information and services. This experiment extends those ideas even further by using

context not only as a factor for providing information or services, but to also use it as the primary source for presenting information, the meeting notes. This functionality is well received by the users, and the very positive responses to statement seven indicate that this push of information is well integrated as well as device discovery and registration are seamlessly weaved in and functioning properly.

Ahmed et al. (2005) present in their work the concept of a Smart Meeting Room that not only detects meeting start and stop, but also provides participant support. The concept of grouping context-aware features according to user clusters is discussed. This work differentiates from this concept by extending the rules by which the cluster is defined by using context-awareness such as proximity, user availability and appointment meta tagging as a basis. From this, one can create ad hoc user clusters in meeting rooms, and for each of these clusters dynamically share meeting information in the form of meeting notes. User responses to statement eight and nine support the interpretation of cloud-based services with personal information, and thereby enable user profile specific context-aware features such as meta-tagging. The responses are not consistent and some indicate that they are not comfortable with this tight integration. Further research on how this can be conducted in an unobtrusive way is needed, especially bearing in mind participants responses to statement ten, where the vast majority is comfortable with their personal information being stored in the cloud.

Cutrell et al.'s (2006) work on calendar integration and the work of context-tags from Göker et al. (2004) are both used as a foundation in my work. My contribution extends this by combining the two approaches and tagging context-aware information in the Google Calendar, as well as by using the Google cloud as the point of integration.

Mei et al. (2008) discussed the behaviour of cloud applications in their work and suggested adaptive behaviour as one possible further research topic. This experiment continues their ideas by adding context-aware dimensions to cloud computing such as meta-tagging and user status computation. The enabling of cloud-based information is an important factor as it eases integration of data in the developed

solution without the need to be concerned about distribution or the scale of the solution.

A challenge identified by Lovett and O´Neill (2010) and Mei et al. (2008) highlights the complexities of cross platforms, cross services gathering and sharing of context-aware information. Although I do not propose a new common definition of context-awareness and how this should be identified and integrated in different settings I do suggest a means for sharing context-aware information. By utilizing protocol buffer for information exchange, a cross device, cross platform and cross service standard for how this can be implemented through contract first approach can be set. This eases the exchange of context-aware information and enables platform and services to seamless integrate with each other. As indicated through the responses in statement 14, multimedia content is in demand. Further research will have to be conducted in order to see how protocol buffer implementations can meet this need in a cross system manner.

5.9.1 Limitations

The Google cloud can be seen as a limitation with this research. The cloud provides several services but at the same time these services are out of reach, opening for server downtime vulnerability. I acknowledge that since some evaluation questions concern personal preferences, such as the one about storing information in the cloud, the results could be somewhat different with a larger population in the evaluation.

5.10 Conclusions

This experiment has featured the context-aware meeting room application. Based on the results obtained guidelines for the two main areas of context-awareness and content and information sharing through cloud integration were identified and discussed. The solution was implemented using the Google cloud application suite. The Google suite was chosen for its included application portfolio including Docs, Calendar and Gmail. Other cloud providers could have been chosen as well, but would have required a different programming interface for service integration. Another argument for using Google cloud was the usage of Android phones as test devices. As already acquainted with the Google eco-system, users of these devices

would already have existing user accounts and would not pose any security issues. Platform as a service providers such as Salesforce (Salesforce, 2012) or Amazon Elastic Cloud (Amazon, 2012) was not chosen for their lack of any application portfolio.

The application suite developed included features for automatic proximity-based recognition of participants, dynamic meeting scheduling and heterogeneous client to cloud integration. Results from the experimental evaluation show that the application was well received by users. As this work extends previous research, these results contribute towards a solution for an adaptive and integrated cloud solution. Interesting novel features such as Bluetooth ping for client discovery and the protocol buffer as message format for heterogeneous cross device / platform message exchange were also incorporated in the work described in this chapter. Returning to the evaluation guidelines by Hevner et al. (2004), an assessment on research output from this experiment is given in the next table (Table 5-3).

Table 5-3 Research experiment mapped to evaluation guidelines

Guidelines	Completion
Design as an Artefact	Considered complete based on the use of models to describe the architecture (use case and class diagrams) and with the developed artefacts (Four mobile clients and two server applications)
Problem Relevance	Considered complete through the identification of the research gap after reviewing existing work in literature review section and through the mapping between research objective and experiment
Design Evaluation	Considered complete by having the artefacts (models) evaluated in practice by code implementation and software testing. The experiment evaluation by 52 users (12 pilot and 40 main evaluation) ensures validity and provides a ground for limited generalization
Research Contributions	As shown through the findings and the research evaluation new knowledge about context-awareness and cloud integration can be extracted from this work. Novel proximity determining functionality in terms of Bluetooth ping expands the context-aware dimension. Also

	expanding this dimension is the heterogeneous message format implementation of protocol buffer. This works expands the ideas of ad hoc user clusters and add novel context-aware features to cloud based applications with user status and calendar meta tagging.
Research Rigour	Considered complete with SCRUM being used as software development methodology and software craftsmanship techniques from eXtreme programming applied for quality assurance.
Design as a Search Process	Considered complete as the Design Research process model by Vaishnavi and Kuechler (2007) has been followed as a framework in this experiment
Communication of Research	Considered complete through the findings summarized in this conclusion and by building upon these results in experiment three (Chapter 6).

This experiment's findings, realizes the second objective: to *explore the integration of context-aware information from cloud-based services and applications in heterogeneous mobile applications* and sets the foundation to establish a suggested cloud to device interplay model. The implementation of Protobuf as cross-platform messaging format, answers one of the issues set forth by Lovett and O'Neill (2010) on how to communicate between frameworks / languages / platforms. User evaluation was performed and provided valuable statistics to be used in evaluation and as ground for further research.

5.10.1 Further Research

Cloud integration has many opportunities for future research. Especially interesting would be to see how one could integrate user preferences and user data in an unobtrusive way. To explore the dynamic relationship between a client device and a cloud server would also be a novel perspective, in order to research further what the right balance would be for a maximum user experience and seamless context-awareness integration. Last but not least, multimedia meeting notes and richer data, as identified by the evaluations of the context-aware meeting room application can be pursued in future work.

Chapter 6

The Context-Aware Phone

6.1 Introduction

Context-awareness has been around for several years and no one doubts its use or importance for creating richer and smarter tailored user experiences. Context-awareness has matured from initial prototype implementations to desktop implementations and most recently to the world of mobile devices and ubiquitous computing. There have also been a few attempts at integrating context-awareness into cloud solutions, but nothing has yet been standardized as this is only at an exploratory stage. On the other hand, cloud computing as an alternative for outsourcing IT solutions has grown popular (Mei et al., 2008; Armbrust et al., 2010). The flexibility and scalability of the cloud facilitates for application tuning by turning on and off hardware according to traffic volume. In the previous chapter, I investigated a context-aware meeting room and exemplified how cloud solutions could be taken advantage of in such a setting. For this experiment I delve further into context-awareness and cloud computing, investigating how the notion and focus of context-awareness can be centred on the user by creating a context-aware phone solution. This addresses the third objective of this thesis: to *create an adaptive user experience by exploiting multidimensional context-aware information in relation to cloud computing on an Android device.* To realize the objective and to achieve the main goals of making interaction between cloud and mobile device as seamless and uncomplicated as possible for the user, several tasks need to be performed:

- Define sources of context-aware information from sensors in phone
- Decide upon client to cloud architecture implementation
- Develop custom push to device messaging

- Implement an adaptive user interface
- Evaluate the solution with real users to validate the user experience and gather feedback on application adaptability

6.2 Literature synopsis

The notion of context-aware computing is generally the ability for the devices to adapt their behaviour to the surrounding environment, ultimately enhancing usability and user experience (Dey and Abowd, 1999). Day and Abowd (1999) have remarked that if we fully understand the context in a given environment and setting we are better able to adapt a context-aware behaviour. The consequence of this is devices that adapt content based on the users' context, eliminating the need for users to manually perform these tasks. Indeed, context is a major part of our daily life and support for sharing and using contextual information will improve user interaction (Edwards, 2005). In my research I combine context information received both from the phone itself and information retrieved from cloud-based servers. This research builds on the experiences of multi-dimensional context-awareness from experiment one (Chapter four) and from cloud-integrated context-awareness from experiment two (Chapter five). All data are integrated to create a context-aware mobile device, and now further extended by a sensor dimension from the phone.

Sensors are an important source of information input in any real world context and several previous research contributions look into this topic. For instance, Parviainen et al. (2006) approached this area from a meeting room scenario. Modern smartphones have a number of built in sensors and they can often be accessed through local interfaces in the operating systems. In our work, by taking the aspect of sensors and context-awareness and integrating it in a mobile application, the workload for the end users is reduced. Thus, the goal of a customizable and adaptable phone that can be configured remotely but also being able to adapt and change during use is met. This creates a new concept of context-awareness and embraces the user in ways previously unavailable.

Through the review of context-aware computing, one highlighted issue is the use of external services as part of the computational foundation. One mechanism for dealing with such a distributed environment is cloud computing. Cloud computing is

an emerging paradigm (Mell and Grance, 2011), and has only in recent years been gaining foothold (Mei et al., 2008).

Mei et al. (2008) have pointed out 4 main research areas in cloud computing that they find particularly interesting namely a "*Pluggable computing entities, data access transparency, adaptive behaviour of cloud applications and automatic discovery of application quality"*. Christensen (2009) looks into some of the issues portrayed by Mei et al. in their work, describing a scenario for creating next generation mobile applications by utilising cloud services. He describes the use of cloud technology to ease off the burden of the mobile clients. The features modern smartphones are lacking include local storage, processing capabilities, and constant information connectivity. Accordingly, smart mobile devices can then potentially exploit context-awareness and sensors to create rich activity contexts in a dynamic interaction with cloud services.

6.3 Experiment Scenario

Every person today has at least one mobile phone. This phone contains all your means for communication:

- it contains your cell phone communication in terms of phone contact and SMS;
- it contains your internet communication in terms of e-mail addresses and accounts;
- it contains your social internet life in terms of i.e. Twitter, Facebook and Pinterest accounts.

The mobile phone also contains all of your applications, those used for leisure, entertainment, communication and work. This fact leads to the phone being your companion and closest friend.

The smartphone device should be capable of exploiting all of these factors and tailor your experience of using the device to your needs. While at work this capacity is realized by having your work time email, contacts and relevant set of applications. When not in a work setting, i.e. enjoying leisure activities the email accounts, contacts you communicate with and your relevant applications are different and

should be dynamically changed. Additionally, the user interface should behave in a smart and adaptable way. The tailoring of information should be adjusted to your current perception of information and be supportive and not a hindrance. An example of this includes adjusting the light setting according to your surroundings. This chapter presents the context-aware phone, an application that adjusts and tailors the user experience to the user's needs. Cloud technology, sensors and push messages are interweaved in the solution creating a seamless experience for the user. I will further detail the contents of this research project, starting with application design and architecture.

6.4 Application Design And Architecture

The developed application consists of three major parts. An Android client used by end users, a server application deployed to the Google App Engine cloud and the Google App domain service suite. An overview is offered below (Figure 6-1).

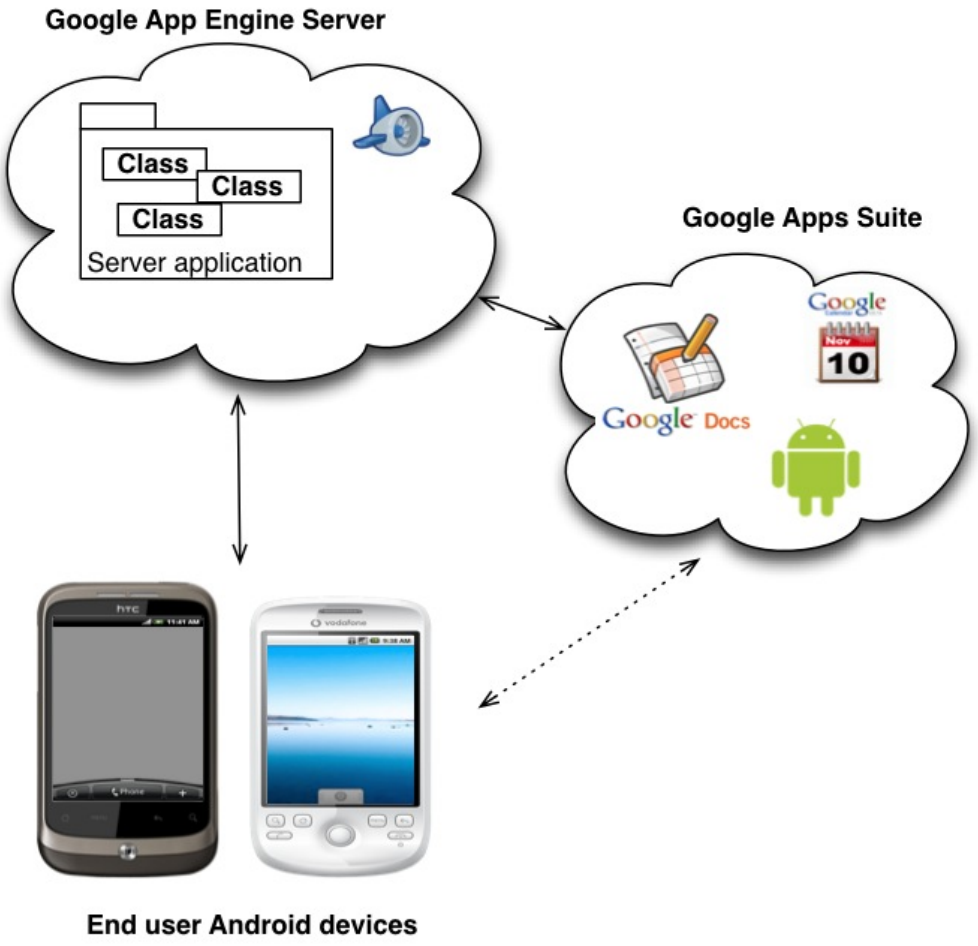


Figure 6-1: System high-level overview

Building on previous Google cloud experience from chapter five, the lessons learned when designing the architecture for experiment two (Figure 5-3) were incorporated. Although the application in this experiment were build from scratch, previous experience with communication protocols and persistence mechanisms from the Google cloud helped speed up the development process. The major difference was the utilization of new Google resources compared with experiment two. Google Docs integration and parsing, as well as Gmail integration, had to be implemented for the first time. The strong focus on server architecture from early on enabled a clear separation of concerns of the different application modules and the solution as a whole was dived into three major components.

These three major components, the Android client, the cloud server application, and the remote Google services realised together the seamless cloud to device application integration. The server application was deployed remotely in the cloud on the Google App Engine, whilst data were also stored remotely in Google cloud services. The initial set-up and configuration of the application were done by the user through a web page interface in the hosted server application. This initial configuration served as the foundation for the system and was regarded as the 'master' configuration if one would need to restore back to defaults at some point. Another reason for having the users entering this set-up on a standard web page, was the benefit of using a large screen and full keyboard / mouse for input. The set-up process is a one-time process, and, once completed, this configuration can be exported to any device associated with the user.

The Google Apps services were an important part of the experiment. Building on the research experiences from context-aware meeting room (Chapter five), Google service cloud integration was taken one step further. Calendar integration had previously been explored and was now extended to a full-scale integration. Additionally full contact integration was performed, allowing for extraction and filtering of contact data. Together with configuration settings, these two dimensions created a rich context-aware foundation from the cloud.

In the Android clients, the context-aware phone application was installed. Once the user had performed identification, the application was registered by the cloud server application and initialized. Given a rich Android device as client, the device capabilities for providing context-aware information were explored further. This consisted of exhaustion of proximity data and sensor integration. These new dimensions were used as input together with cloud server information to create a combined context-aware foundation upon which the user context was computed. Based on this, action was taken and rich, tailored, user experiences produced.

6.4.1 Architecture

Further I will do a detailed presentation of the system architecture, highlighting novel points and justify the architectural design decisions. Figure 6-2 gives an overview of the implementation of the system. Blue boxes in the diagram represent the parts of the system I created and the white boxes represent external systems that were communicated with, like Google Calendar and Contacts.

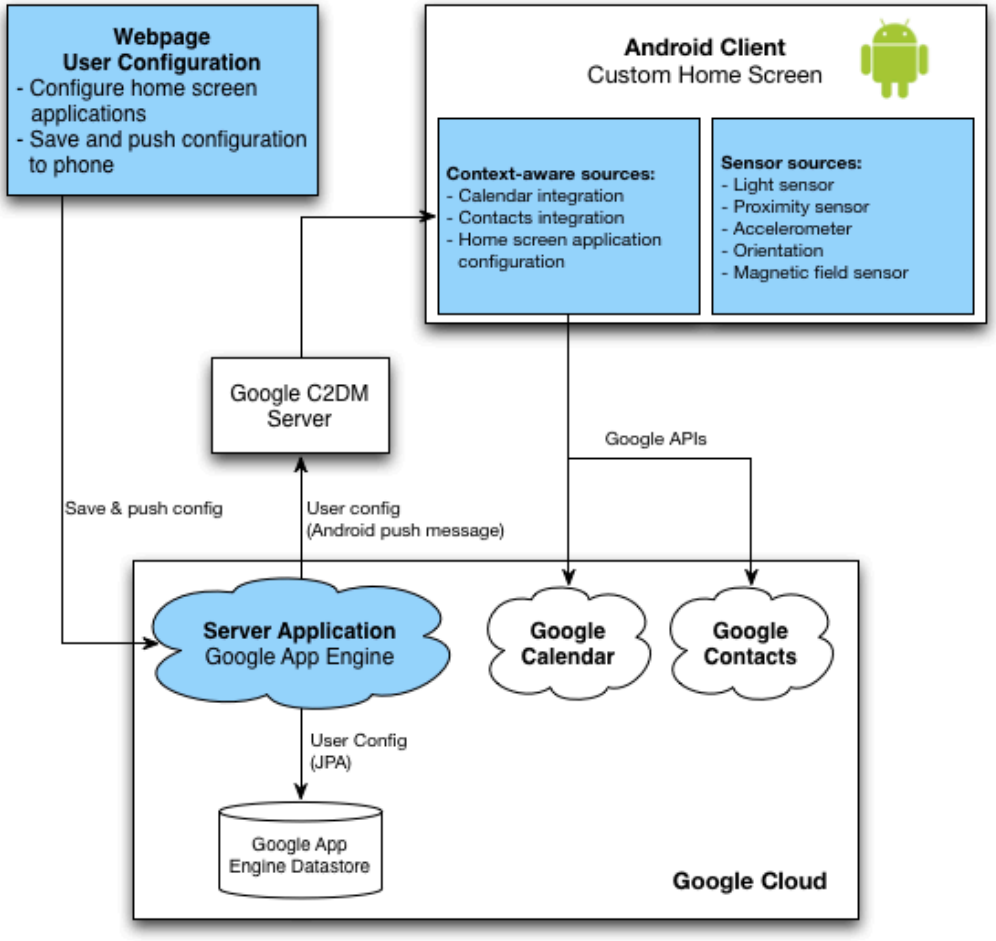


Figure 6-2: System Component Architecture

6.4.1.1 Server application

The main purpose of the server application was twofold. Firstly, it should handle all communication with the clients in regards of their initial set up. Letting the application host a web module serving a registration page takes care of this. The clients would start off by accessing this web page and log in using their Google identity. By implementing Open Authorization (OAuth) client data could be extracted and integrated into this application without storing their credentials locally, thereby avoiding creating possible malicious interests. OAuth uses access tokens for authentication and can as a third party library access services and resources on the users' behalf. After the clients successfully authenticate themselves on the web page they are greeted and taken to the configuration page, Figure 6-3.

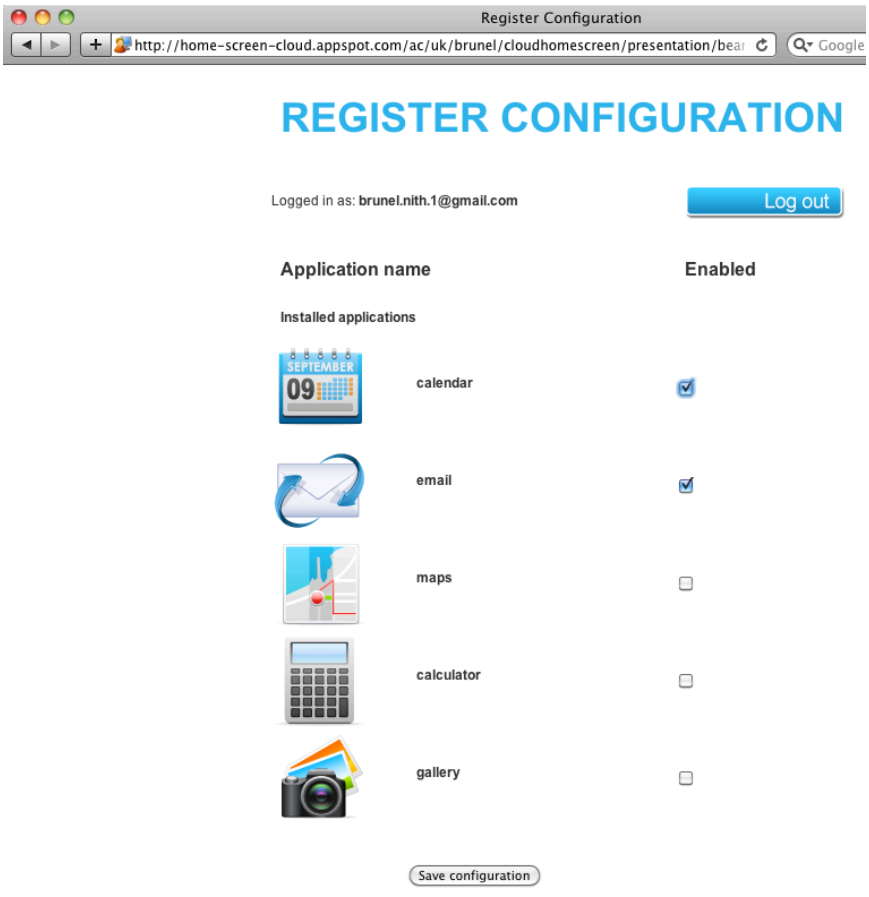


Figure 6-3: User Registration and Configuration Page

From this configuration page, the user can do an initial set-up of their device and save it as default. When the user saves the configuration, this is pushed to the user's device and the phone will then update its user interface. This configuration gave

benefits such as not tying a configuration to a device, but rather to an individual user and ease the restoration of default settings.

6.4.2 Google Cloud Services

This experiment utilized the Google App Engine as a cloud backend server. This allowed a focus on *data access transparency*, where clients transparently will push and pull for data from the cloud, and *adaptive behaviour of cloud applications*. The behaviour of the Google App Engine server application was adapted based on context information sent from the users' devices, thus integrating context and cloud on a secure (Mowbray and Pearson, 2009) mobile platform. From the deployed server application, I integrated the Google Apps Calendar, Contacts and Gmail. To make it possible for users to tag their appointments and contacts with context information meta-tags were added. By adding a type tag, for example *$[type=work]* or *$[type=leisure]*, it was possible to know if the user had a business meeting or a leisure activity. The contacts were then filtered based on this information. If the tag *$[type=work]* was added, this lets the application know that the user is in a work setting and it will automatically adapt the contacts based on this input. In a work context, only work related contacts would be shown. To add and edit these tags the web-interface of Google contacts and calendar was used.

6.4.3 Android client

The client application was implemented on an Android device. This application utilized context-aware information from the device in the form of time, location and sensors. Additionally it utilized context-aware information from the cloud-integrated backend to acquire dynamic interface content, contacts and calendar data. At launch, the application would look as illustrated in Figure 6-4.

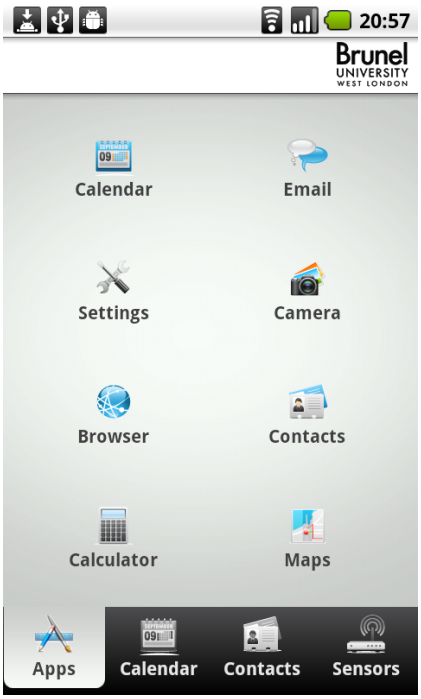


Figure 6-4: Home screen interface at launch of application

This interface would change depending on the user's given context. The applications available would be adapted and customized to match the current computed user context and thereby, unobtrusively alter the user experience.

The sensors on the mobile device were also used as input to the application. The API available on the Android platform was used and through a base class called *SensorManager* all of the built-in sensors on the mobile device were accessed. The pilot study started off just by showing the input value from the sensors, but for the expanded version of the application values were incorporated into the application. When expanding the application after the initial tests more sensors input were used to further enhance the user experience. The final version ended up using two features directly in the application, the accelerometer and the light sensor. The accelerometer was used to register if the device was shaking. If the device is shaking it means that the user is probably on the move, for example running or walking fast. In these cases, an automatic change of the user interface to a much simpler view that has bigger buttons and is easier to use when on the move, was triggered.

The second sensor used in the experiment was the light sensor. By constantly registering the lighting levels in the room / the environment it gradually adjusted the

background colour of the application. The very careful adjustments was due to the fact that it would be very annoying for users if colour changes occurred often and were too drastic. Figure 6-5 visualizes how this looks, showing lighting levels for two images with significant differences. The images are from the sensor administration panel (Tab 4 in Figure 6-4), whereupon after the prototype evaluation was also integrated in the application user interface.

Figure 6-5: Light sensor data output screens

6.4.4 Push to Device Messaging

The messages from the server application on the Google App Engine are sent with a push messaging feature for Android called C2DM (Cloud to Device Messaging), available from Android 2.2. The C2DM feature requires the Android clients to query a registration server to get an ID that represents the device. This id is then sent to our server application and stored in the Google App Engine data store. When a message needs to be sent, the "save configuration"-button is pushed. The message was composed according to the C2DM format and sent with registration id as the recipient. These messages were then received by the Google C2DM servers and finally transferred to the correct mobile device. The process is illustrated below (Figure 6-6).

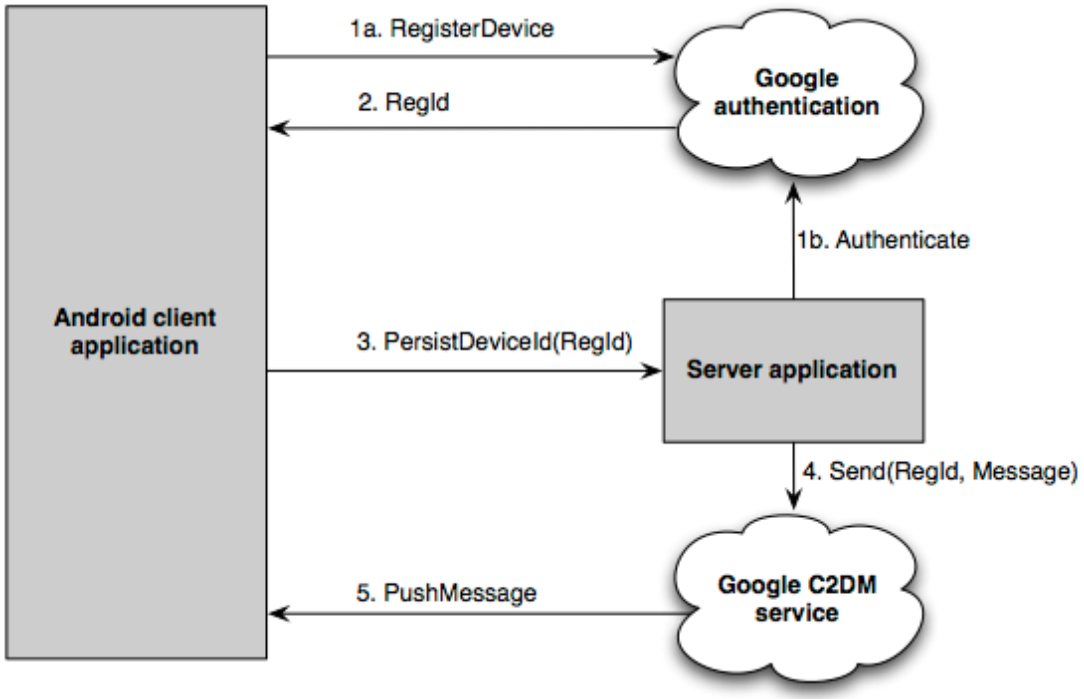


Figure 6-6: Cloud to Device Messaging Overview

The technology is used in several standard Google applications including Gmail, Contacts and Calendar. The message limit is set to 1024 bytes and developers are encouraged to send short messages, essentially notifying the mobile application that updated information can be retrieved from the server. Google offers standard libraries for Android that makes it possible to use C2DM directly. In this experimentation with standard C2DM, the API and tools were found to be very limited and in certain cases it provided a low level of abstraction. To make it easier to use C2DM without the need for a commercial third-party library a customized version called Simple-C2DM was implemented as a wrapper around the C2DM library, the aim of which was to try to both increase the abstraction level and simplify tasks that are unnecessarily complex and in some cases even avoidable, such as manual operations.

6.5 Participants

The research experiment had two main phases of evaluation. In phase one, a pilot test was performed with a total of 12 users. These users were of mixed age, gender and computer expertise. The results from this phase were fed back into the development loop, as well as helped remove some unclear questions in the questionnaire. In the

second phase, the main evaluation, another 40 people participated. Out of the 40 participants in the main evaluation, two did not complete the questionnaire afterwards and were therefore removed making the total number of participants 38 in the main evaluation. All 12 from the pilot test and the 38 from the main test session were aged between 20 and 55 years old. All participants had previous knowledge of mobile phones and mobile communication, but had not previously used the type of application employed in our experiment. None of the pilot test users participated in the main evaluation. From the user computer experience classification (asserted based on a questionnaire employing the taxonomy of McMurtrey (2001) I learnt that the majority of the users had a good level of computer expertise.

6.6 Material

The material used in this test consisted of three different mobile devices (Table 6-1). They were chosen as each of them represents the Android operating system and they contained a sufficient selection of sensors.

Table 6-1: Test Material

Device	Operating System	Used for
HTC Nexus One	Android	Customized phone tests
HTC Evo	Android	Customized phone tests
Samsung Galaxy Tab 10”	Android	Gathering push test data
Apple MacBook Pro	Os X	Development

For development hardware two MacBook Pro 2.4 GHz Intel core 2 Duo machines with 4 GB of RAM were used. The main tasks of these machines were to run the development environment. In addition to this, the Google App Engine cloud server was used as the backend server application.

6.7 Procedure

The main user evaluation was accomplished by having 40 people attend different test session and try out the application and functionality as described. They were initially told to start the context-aware phone application on the smartphone device they received. By entering a given Google account credentials they were taken in to the application. They then familiarized themselves with the interface, also going through the tabs for contacts and calendar and playing with the sensors.

Afterwards, they were told to enter a given URL taking them to the Google App Engine and the server application. After logging in, they performed an initial configuration of the device and had this configuration pushed to their test device. Back in the Android application they were told to play more around with the device, seek places with more or less light to trigger the light sensor as well as using the app while moving to trigger the "on-the-move" version of the interface. Finally they played around with contacts and appointments and were asked to observe how the phone interfaces adapted accordingly. The test ended with the candidates answering an online questionnaire (Please refer to Appendix E). The whole test procedure was handed out to the participants at the start of the test (Attached in Appendix F).

6.8 Results

The results presented here illustrate different parts of the questionnaire: statements one to three target the *user interface*, statements four to six regard *sensor integration*, statements seven to nine focus on *the web application*, statements ten to thirteen centre on *context-awareness,* while statements fourteen to seventeen are about *cloud computing*. The questionnaire ends with *overall usefulness,* which is in an open-ended question for comments. The statements are given below (Table 6-2) together with the mean, standard deviation and the results of applying a one-sample t-test.

Table 6-2: User evaluation questionnaire and results

	Statement	Mean	Std. Dev.	t-test
Statements regarding user interface				
Statement 1	It is easy to see the available functions	3.50	0.51	**.000**
Statement 2	The features of the application are hard to use	1.89	0.73	.378
Statement 3	The adaptability of the application is a feature I approve	3.45	0.55	**.000**
Statements regarding sensor integration				
Statement 4	The background colour in the application changes when the lighting in the room changes	3.55	0.65	**.000**
Statement 5	When moving around, a simplified user interface is not presented	2.11	1.06	.544
Statement 6	I find sensor integration annoying and would disable it on my device	1.84	0.72	.183
Statements regarding the web application				
Statement 7	I was able to register my device application configuration in the web application	3.61	0.59	**.000**
Statement 8	I was not able to store and push my configuration to my mobile device from the web page	1.47	0.80	**.000**
Statement 9	I would like to configure my phone from a cloud service on a daily basis, (webpage user config and Google services like mail/calendar/contacts)	3.18	0.69	**.000**

Statements regarding context-awareness				
Statement 10	The close integration with Google services is an inconvenience, I am not able to use the system without changing my existing or creating a new e-mail account at Google	1.76	0.88	.107
Statement 11	Calendar appointments displayed matched my current user context	3.58	0.55	**.000**
Statement 12	The contacts displayed did not match my current user context	1.29	0.52	**.000**
Statement 13	I would like to see integration with other online services such as online editing tools (for example Google Docs) and user messaging applications (like Twitter and Google Buzz)	3.29	0.73	**.000**
Statements regarding cloud computing				
Statement 14	I do not mind Cloud server downtime	2.08	0.78	.539
Statement 15	I do not like sharing my personal information (like my name and e-mail address) to a service that stores the information in the cloud	2.16	0.79	.225
Statement 16	Storing data in the Google Cloud and combining this with personal information on the device is a useful feature.	3.26	0.60	**.000**
Statement 17	I find the cloud-to device application useful	3.53	0.51	**.000**
Overall satisfaction				
Open comment question				

6.8.1 User interface

Statements one to three deal with the user interface (Figure 6-7). These results reveal positive facts about the interface, highlighting that the majority of the users found it easy to see all available functions; moreover the vast majority (37/38) also finds the adaptability of the application a feature that they approve of. However, looking at statement two, their opinions are split in terms of whether the features are hard to use, and the statement results are statistically not significant.

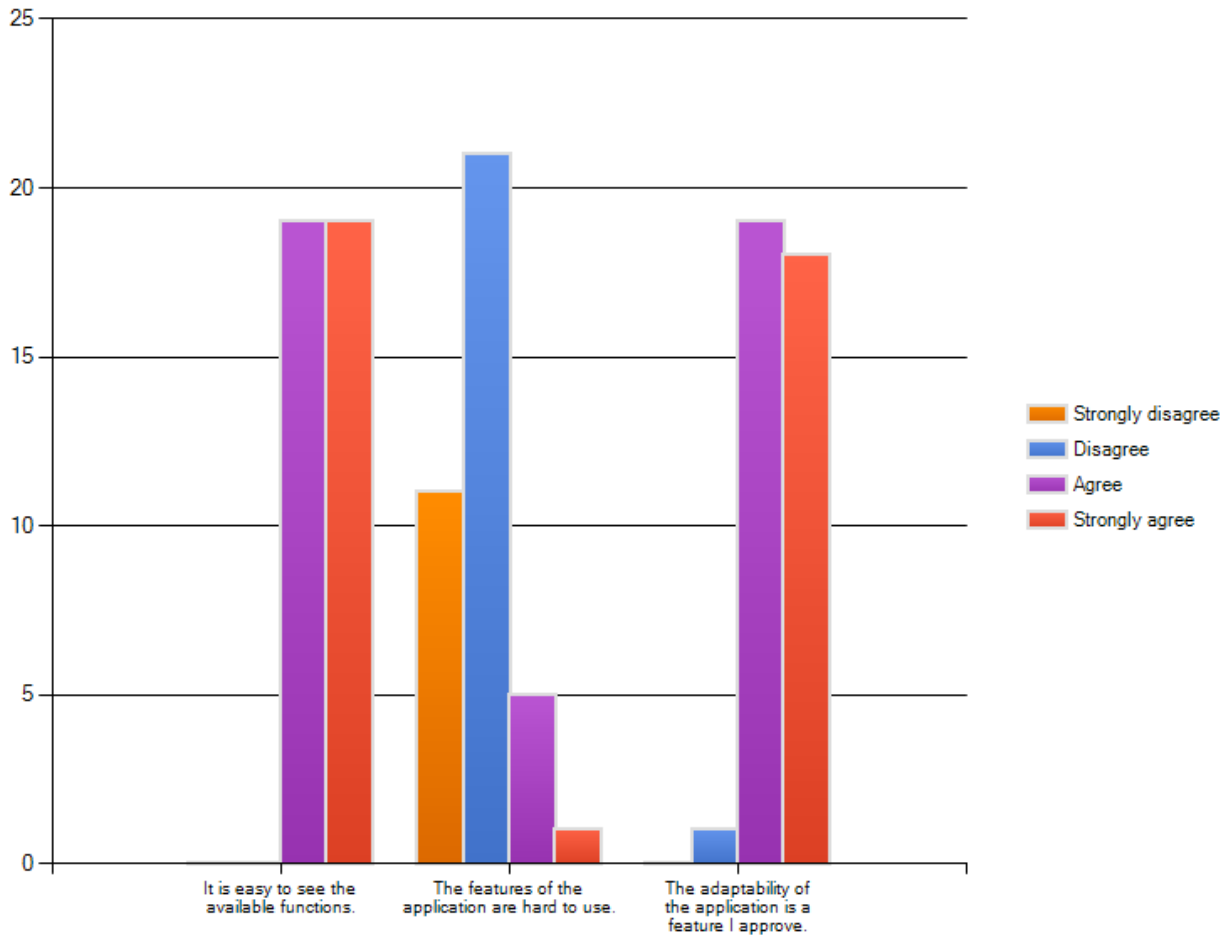


Figure 6-7: Statements regarding user interface

The results indicate that it is easy to get an overview of the application and the test candidates find adaptability a positive feature.

6.8.2 Sensor integration

Opinions are split regarding sensor integration. Users agree that the light sensor is working as expected, but disagree whether the simpler user interface changes. This can be due to the sensitivity threshold programmed for the sensor, and should be

verified by more comprehensive testing. The majority, 32 out of 38, would not deactivate sensor integration and this is a useful observation, highlighting that sensors should be further pursued as context-aware input.

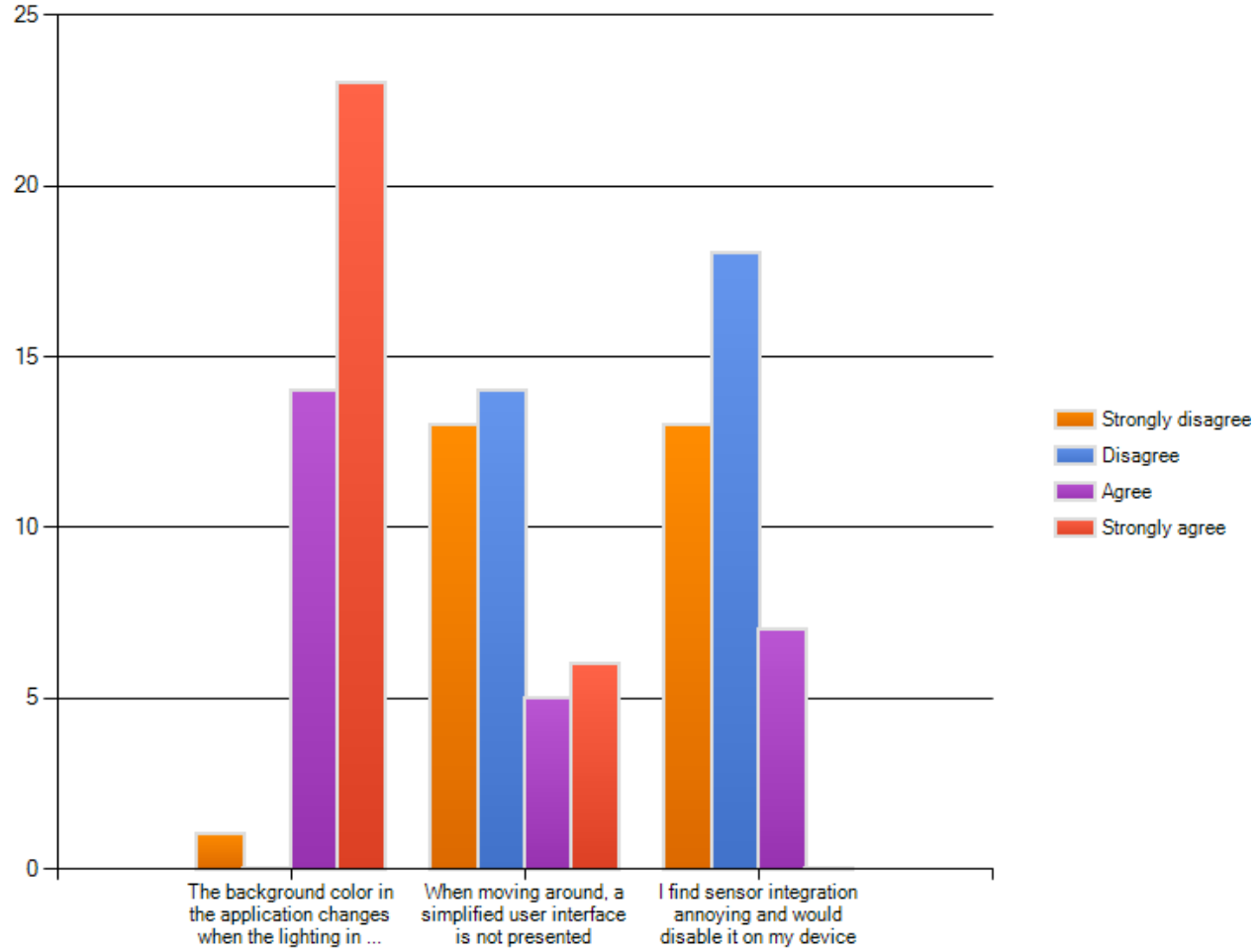


Figure 6-8: Statements regarding sensor integration

6.8.3 Web application

The statements dealing with the web application at Google App Engine (Figure 6-9) show that the web application performed as expected, by letting participants register their devices as well as pushing performed configurations to the devices. Also answers from statement nine are quite interesting, highlighting a positive attitude towards cloud-based services (32 out of 38 are positive).

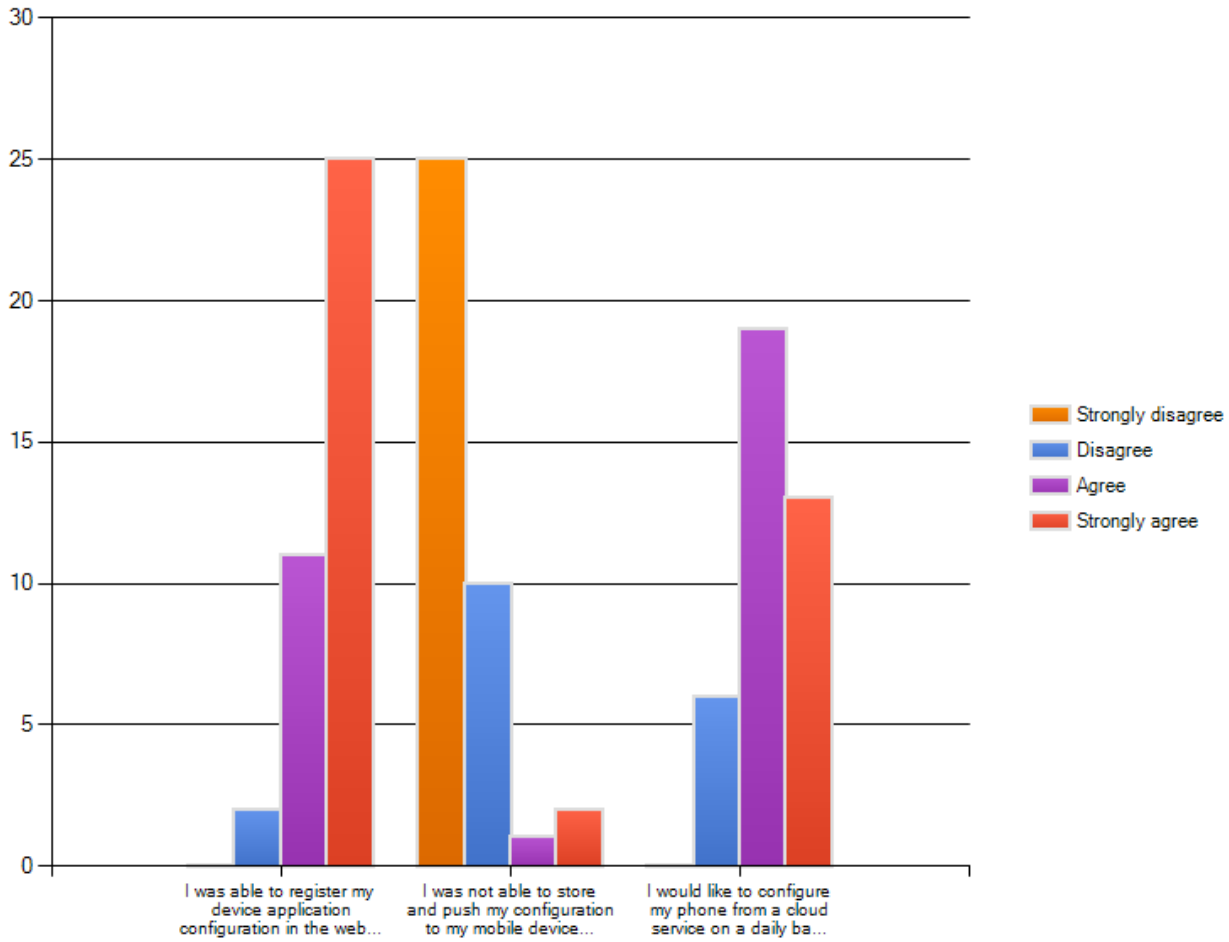


Figure 6-9: Statements regarding the web application

6.8.4 Context-awareness

In terms of context-aware information the participants were asked to take a stand in respect of four statements, with results shown below (Figure 6-10). For the first statement (S10), although a clear majority supported this assertion (33/38), opinions are somewhat spread and this answer is not statistically significant. For the next two questions a very positive bias are shown, indicating correctly computed context-awareness and correct presentation to the users. Again for statement 14, users are eager to see more cloud-based services and integration.

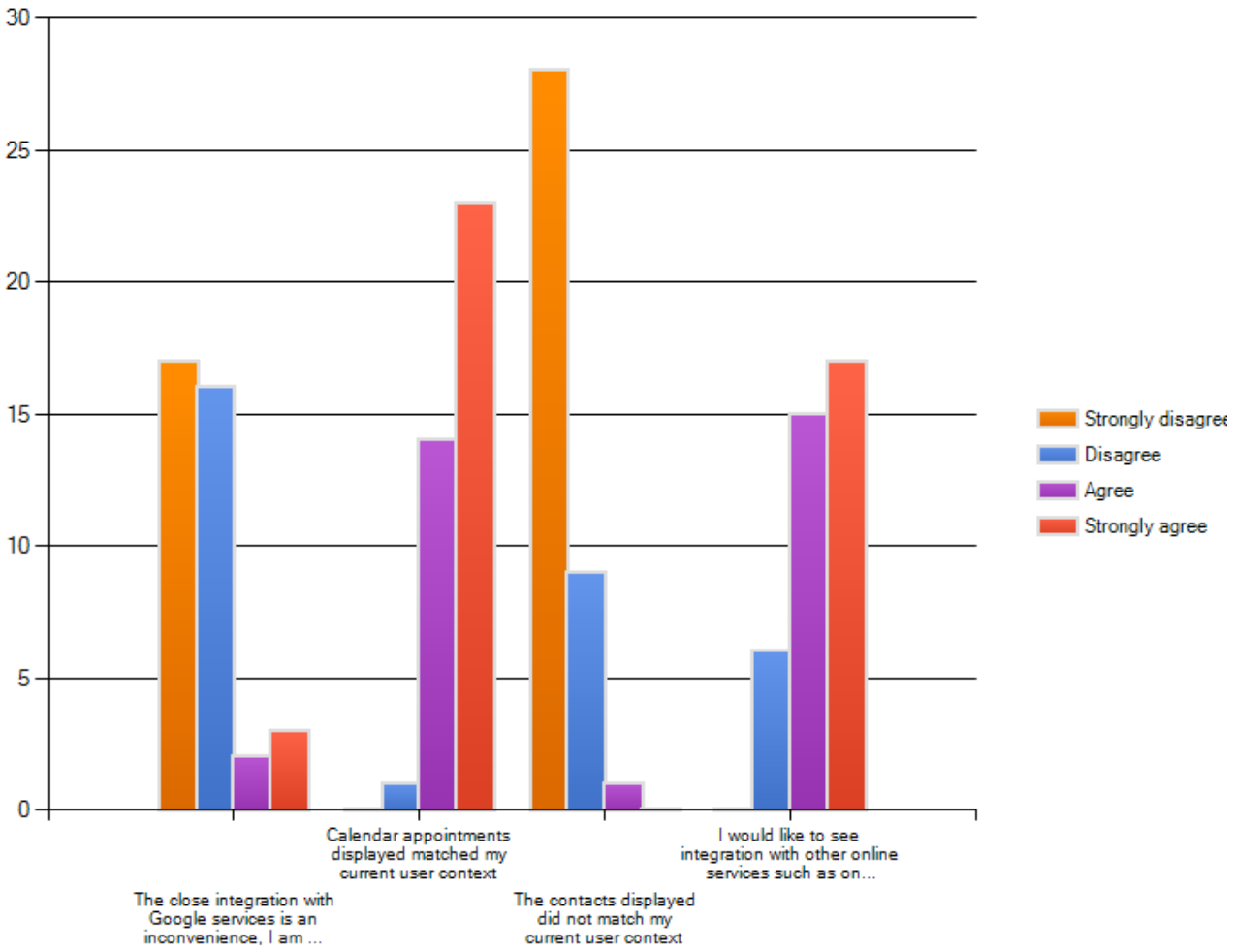


Figure 6-10: Statements regarding context-awareness

6.8.5 Cloud computing

When inspecting results from the cloud-computing section, results are mixed and differences in opinions do occur. For statements 14 and 15 the results are not statistically significant, but they indicate a mixed attitude towards cloud vulnerability and cloud data storage. The two statements with statistically significant results, statement 16 and 17, participants find storage of data in the cloud and using this as part of the data foundation for the application a useful feature and are positive towards it. Their answers also suggest a fondness for push based application configuration.

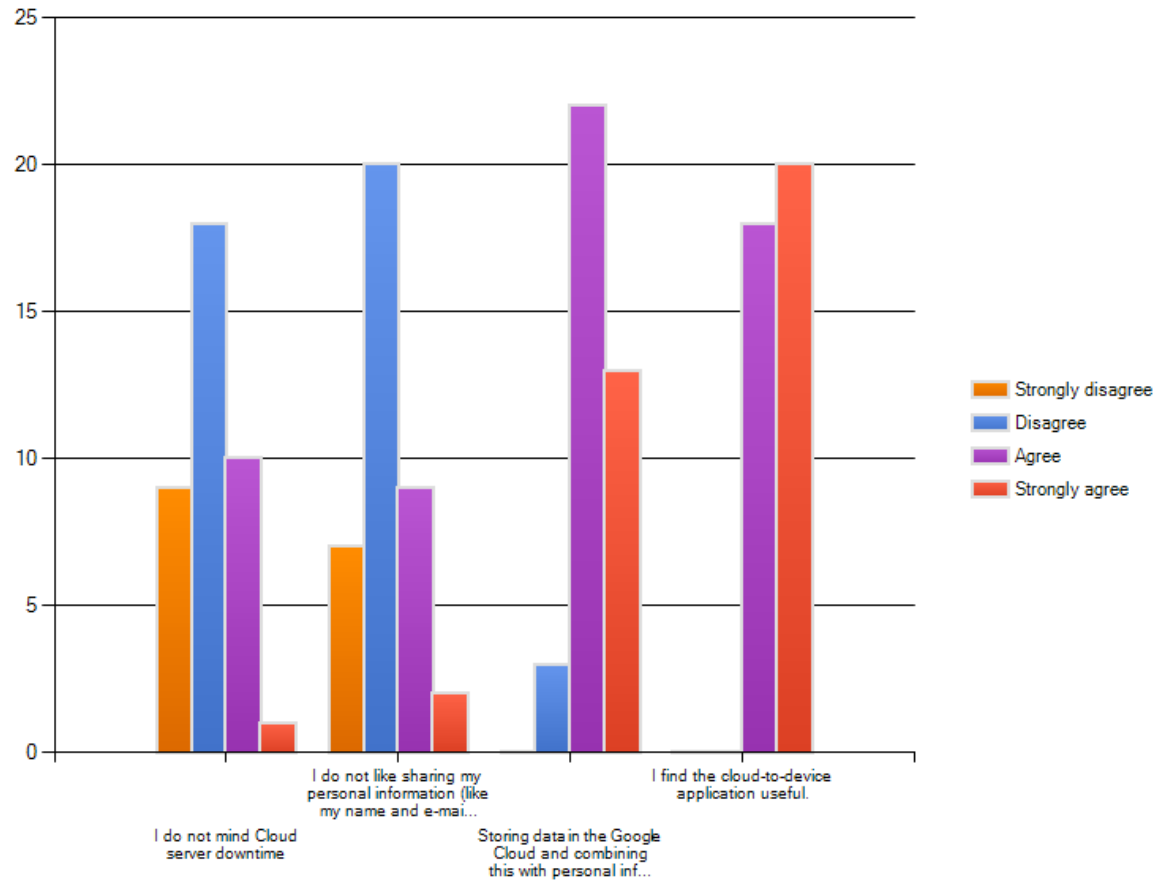


Figure 6-11: Statements regarding cloud-computing

6.9 Discussion

From the literature review and the summary in this chapter I point at the ability for modern applications to adapt to their environment as a central feature (Dey and Abowd, 1999). Edwards (2005) argued that such tailoring of data and sharing of contextual information would improve user interaction and eliminate manual tasks. Results from the user evaluation support this. The user found it both attractive as well as have positive attitudes towards automation of tasks such as push updates of information by tailoring the interface. This work has further elaborated on context-aware integration and shown how it is possible to arrange interplay between on device context-aware information, such as sensors, and cloud-based context-aware information such as calendar data, contacts and applications. These sources are followed up by suggestions for further research on adaptive cloud behaviour as identified by Christensen (2009) and Mei et al. (2008).

To register the tags the standard Google Calendar and Contacts web-interface was used. Such a tight integration with the Google services and exposure of private information was not regarded as a negative issue. As shown in Figure 6-11, most of the users surveyed disagreed that this was an inconvenience. This perception makes room for further integration with Google services in future research, where, amongst them, the Google+ platform will be particularly interesting as this may bring opportunities for integrating the social aspect and possibly merge context-awareness with social networks.

Sensors are an important source of information input in any real world context and several previous research contributions look into this topic. The work presented in this chapter follows in the footsteps of research such as that of Parviainen et al. (2006), and extends sensor integration to a new level. By taking advantage of the rich hardware available on modern smartphones, the developed application is able to have tighter and more comprehensively integrated sensors in the solution. From user evaluation one can learn that although sensor integration as a source for context-awareness is well received, there is still research to do. In particular this has to do with the fact to what extent what thresholds should be used for sensor activation and deactivation. It has been shown that it is feasible to implement sensors and extend their context-aware influence by having them cooperate with cloud-based services. Further research should investigate sensor thresholds and additionally see if there are differences in people's perceptions of different sensors.

As previously mentioned, the state-of-the-art feature C2DM was used to push messages to the mobile devices and a custom version, Simple-C2DM, was implemented, to overcome some obstacles in the original implementation

6.9.1 Limitations

This experiment has a small evaluation sample and further research should be performed in order to generalize the results. The test was performed on two different Android smartphones, and variations in behaviour may be an issue when it comes to other vendor implementations. Being a single platform solution is also a limitation

and further research should consider including multiple platforms and operating systems.

6.10 Conclusion

This research project has added a new and novel contribution to the area of context-awareness. A scheme for harvesting context-aware information from several dimensions including on device, cloud and user centric, forms a rich basis for algorithms for context-aware computation. Furthermore, this effort has shown a novel approach for tailoring the user experience by utilizing push to device messaging in cooperation with user context. Through the developed applications a possibility for automatic configuration and adaptation is shown. Future research should continue to innovate and expand the notion of context-awareness enabling further automatically adaptation and behaviour altering in accordance with implicit user needs. Returning to the evaluation guidelines by Hevner et al. (2004), an assessment on research output from this experiment is given in the next table (Table 6-3).

Table 6-3: Research experiment mapped to evaluation guidelines

Design Research Guidelines	Completion
Design as an Artefact	Considered complete based on the use of models to describe the architecture (architecture and component diagrams) and with the developed artefacts (Android client and server application)
Problem Relevance	Considered complete through the identification of the research gap after reviewing existing work in the literature review section and through the mapping between research objective and experiment
Design Evaluation	Considered complete by having the artefacts (models) evaluated

	in practice by code implementation and software testing. The experiment evaluation by 50 users (12 pilot and 38 main evaluation) ensures validity and provides a ground for limited generalization
Research Contributions	As shown through the findings and the research evaluation new knowledge about context-awareness and cloud integration can be extracted from this work. Novel ideas for combination of: sensor, push technology and cloud services are the basis for creating a tailored and adapted user experience. Additionally this research contributes with a custom version of the push to device messaging framework for Android, Simple-C2DM.
Research Rigour	Considered complete with SCRUM being used as software development methodology and software craftsmanship techniques from eXtreme programming applied for quality assurance.
Design as a Search Process	Considered complete as the Design Research process model by Vaishnavi and Kuechler (2007) has been followed as a framework in this experiment
Communication of Research	Considered complete through the findings summarized in this conclusion and by building upon these results in the thesis conclusion (Chapter 7).

This experiment findings realizes the second objective: to *create an adaptive user experience by exploiting multidimensional context-aware information in relation to cloud computing on an Android device,* with the goal of establishing and assessing the impact of a real world, full-scale context-aware experience on a mobile information platform. By utilizing research results from multi-dimensional context-awareness and from cloud integration a new adaptive user interface for content tailoring on a mobile phone was developed.

Chapter 7

Conclusion

7.1 Area of Research

The use of context-awareness as a source of information to create more intuitive and feature-rich user experiences on smartphone devices is the driving factor behind the work described in my thesis. This evolvement of the user experience comes from a desire of utilizing as much resources as possible from smartphone devices. Considerable amount of work has previously been done in context-awareness, with Dey's (2001) broadened characteristic of the field as the initial, modern, milestone. The issue with existing research is that most approaches consider only one, or just a few sources of context-aware information. Works have looked into cognitive sides of context (Prekop and Burnett, 2003), user status (Schmidt et al., 2001), interactions (Zhou et al., 2006) and location (Häkkilä and Mäntyjärvi, 2004) to name but a few examples. Context-awareness has been applied in many settings, with hospitals (Munoz et al., 2003; Gkonis et al., 2011), meeting environments (Ahmed et al., 2005; Chen et al., 2004) and tourism (Ferris et al., 2010; Cheverst et al., 2000) being a few examples. Context-awareness is about taking advantage of the current user situation (Dey and Abowd, 1999).

Although smartphones' performance is continuously improved and hardware evolved, they do fell short when the demands get high enough. The answer to this is to apply the methodology from the world of desktop and enterprise solutions, by utilising cloud computing (Mei et al., 2008). In cloud computing issues such as scalability, security and hardware/software abstraction are taken care of for the user by middleware. Context-awareness is today a common part of mobile applications, and the observation to be made through the literature is the lack of multi-dimensional context-aware applications and the integration with such. The majority of context-aware examples utilize only a small portion of the available contexts and there are

also openings for a stronger integration from cloud computing services into mobile devices. Pluggable entities, data access transparency, adaptive behaviour and automatically assessment of application quality are all identified as important challenges (Mei et al., 2008).

7.2 Research Aim and Objectives

The research aim defined for this thesis was:

> *"To investigate multi-dimensional context-awareness in collaborate interplay with cloud-based mobile services, in heterogeneous smartphone applications to create adaptive user experiences"*

To realize this research aim, three major research objectives were defined. These three objectives were realized through experiments as detailed in each of the experiment chapters. Each experiment contained detailed goals / tasks to answer the relevant objective and through implementations and evaluations findings were identified and contributed towards answering the research objectives and produce the research contributions. Accordingly, I will review the different objectives before moving on to describe the findings and research contributions of my thesis.

The thesis started off by examining the implementation and exploitation of multi-dimensional context-awareness in mobile devices. The use of multiple sources for context-aware information has been identified, as an important factor for creating adaptive solutions able to tailor information to users needs. To ensure a full, comprehensive out of box experience evaluation I followed industrially designed experimental methodologies (as detailed in chapter three). Accordingly, the first objective of the thesis was to:

> *Implement and integrate multidimensional context-aware functionality in a solution for a mobile personal information manager.*

The link between cloud based services and context-aware information has been highlighted as an important research subject and this was assessed from a practical perspective in the intelligent meeting room scenario. A cross platform context-aware application with cloud integration was developed, and then this was evaluated in light of multiple platforms and operating systems. Work in this respect was described in chapter five. Accordingly, the second objective of the thesis was to:

> *Explore the integration of context-aware information from cloud-based services and applications in heterogeneous mobile applications.*

After investigating multidimensional context-awareness, as well as cloud integration in a heterogeneous environment, I further sought to assess the impact of a real world, full-scale context-aware experience on a mobile information platform. By utilizing research results from multi-dimensional context-awareness and from cloud integration, a new adaptive application suite for user interface and content tailoring on a mobile phone was developed. The work in this respect was detailed in chapter six. Accordingly, the third objective of the thesis was to:

> *Create an adaptive user experience by exploiting multidimensional context-aware information in relation to cloud computing on an Android device.*

To ensure validity of the findings and a limited generalizability a rigid research methodology was applied during the project. The Design Research methodology (Hevner et al., 2004) was the chosen approach and the execution followed the steps described by Vaishnavi and Kuechler (2007). During development of artefacts, state-of-the-art techniques from software development were applied to my work. Principles from eXtreme Programming (Beck, 1999) governed the development strategies and ensured that high quality craftsmanship were applied. For administration of tasks and delivery of business value i.e. project artefacts, this work conformed to the principles of the software development framework SCRUM (Schwaber and Beedle, 2002). For rigorous research-based evaluation, the Design Research methodology was combined with the seven guidelines from Hevner (2004).

These proposed guidelines were used as the framework for evaluation of the research output of this project and for each experiment answering a particular objective and for the overall thesis answering the research aim.

7.3 Findings and Research contributions

The main contribution of this work originates from the defined research aims and objectives. Accordingly, in my work I have investigated tailoring of user experiences by investigating areas such as context-awareness and cloud computing in a mobile environment.

The contribution in this work comes from three different areas. The first one has to do with technology enhancements in terms of Bluetooth ping, the second area has to do with cloud computing integration from mobile devices and the third and final area has to do with multi-dimensional context-aware integration and manipulation in mobile devices. I will further detail and review the specific research contribution from all three areas, starting off with – Bluetooth ping.

Bluetooth ping realizes a need identified when connecting with several Bluetooth devices. In the case of using Bluetooth for client identification, in, for instance, proximity based services; a usual approach is to connect to the device by pairing. This process involves a considerable amount of time to be completed and is very inefficient, especially if the client attempted to pair with is not a desired user of the system and thereby rejected. In the case of the meeting room scenario, with potentially a large number of participants, this technique will not scale well enough to actually be used as an identifying technique in a busy environment. To address this issue, Bluetooth ping was presented, which take the area of client identification further by utilizing proximity information to a new level. The concept of the approach was to beforehand be in possession of a list of all accredited participants Bluetooth ids. When scanning for relevant participants a formal pairing would not be performed straight away, but rather the always open and accessible hands free port service of the mobile devices would receive a network ping. If answer were to be received, then a valid participant is identified and a normal pairing process is

initiated. This approach can revitalize the use of Bluetooth as a mean for proximity identification for future research projects.

The second area has to do with *cloud computing integration from mobile devices*. I will address this contribution from two angles. Firstly I would like to highlight a novel extension of existing techniques and tools for cloud to device messaging. Then I will return to the research contributions in regards of cloud-based, context-aware computation.

Push messaging has become an alternative to the older, and more common approach of pull messaging. Both efforts are about delivering updated information to a client, this being an application or a user. In research experiment three (Chapter six) a cloud to device messaging approach was presented. Iterating over Google's implementation of push messaging an implementation that wraps Google's library in an improved version was developed. The major disadvantages in Google's approach included single inheritance, lack of exception handling and cumbersome implementation routines (Google, n.d.-b). In my approach I applied state of the art software engineering craftsmanship techniques such as annotations for easier implementation, refactoring abilities for separation of concerns and the ability to implement checked exceptions. I also improved corresponding configuration files for easier library management. Through performance tests and implementation in a user evaluated experiment it was showed that the developed implementation does not degrade the performance of the library compared to the original and it improved the ease of use for developers. These results help move push to device messaging one step further towards a developer friendly, easy accessible, implementation.

For the second part of *cloud computing integration from mobile devices* my research investigated research combining the areas of cloud computing and context-awareness. Building on experiences from research experiment one, I was able to move the concept of keyword-taxonomy and meta-tagging into the cloud domain. The Google App Engine server was utilized as the deployment environment. From this server I was able to integrate seamlessly, from a user perspective, with several Google data sources such as Gmail, Calendar, Documents and Contacts. To exploit

the services for context-aware computation the concept of meta-tagging was applied. For instance, the calendar description field got extended by a particular entry specifying meta-information about this appointment, and the same approach was applied for contacts. This enabled the solicitation of context-aware logic server side in a cloud application. It is my belief that within the area of context-awareness, which argues for a closer implementation and communication with the field of cloud computing, this is a research increment. User evaluations confirmed the feasibility of the approach and highlighted a way to implement two separate aggregated sources of context-aware information, which, when combined, create an even stronger data foundation for the system to make weighted context-aware decisions on and to be exploited further by the mobile application.

The third part of the research contribution dealt with context-aware integration and manipulation in mobile devices. Identified research gaps highlighted the feasibility of applying multi-dimensional context-awareness in mobile applications, as the common approach today is to use a single/few dimensions of context-awareness. I utilized traditional context-awareness as user location and activity but improved upon earlier solutions. For location, the previously described Bluetooth ping technology was implemented and for activities a local taxonomy was created. Additionally sensors for light, orientation, acceleration and gyro meter were integrated as sources of context-aware information. Building upon research results from experiment one and two this third experiment was conjoined efforts from the earlier approaches. When combining information from all these sources of context-aware information, a rich foundation was created on which decisions could be made.

Such a multi-dimensional approach to context-aware information greatly enhances the tailoring and adaptation possibilities for the applications. Further these results for tailoring and adaptation were combined with the results obtained from experiment two with cloud integration. The full user context computed by the application consisted of multi-dimensional context-aware information from the user's mobile device joined with context-aware information extracted from the cloud. This rich data foundation was in user-evaluated experiments implemented and tested to provide data for considering feasibility and viability of the approach. It is my belief

that the proclaimed solution has possible implications on exploitation of future context-aware solutions, especially when it comes to combining context-aware results from local computations with those of cloud based services and provide users with options for tailoring and adaptation based on the collective efforts of the two.

The research methodology followed throughout this thesis was the Design Research. By applying the different steps as laid out in the methodology chapter, a more structured and rigorous approach has been performed. The use of Design Research has focused on a business anchoring for the problem domain. This leads to this research being grounded both in academic sources as well as in a business domain, coping with a relevant business challenge. Having used Design Research both on micro level, in experiment 1 – 3, and on macro level, for the whole thesis, creates a two-layered approach with full utilization of the methodology and this allows the research to follow a structured form. One major issue with Design Research is the lack of detailed evaluation criteria.

Returning to the guidelines from Hevner (2004) for applying rigorous research-based evaluation, the seven guidelines will in the following table (Table 7-1) be used to summarize the research work and evaluate the output at a general level for this thesis.

Table 7-1: Thesis output evaluation structured by Hevner (2004) guidelines

Design Research Guidelines	Completion
Design as an Artefact	This has been given considerable amount of attention and all three research experiments have focused on providing research outputs (artefacts), which are measureable and concrete. All three experiments have produced source code for applications suites and office based documentation.
Problem Relevance	The relevance of the problem has been addressed through the conducted literature review and are also confirmed by the research community as articles on the work conducted in the research experiments are published (Please refer to publication

	list in cover pages).
Design Evaluation	Considered achieved by applying rigorous testing and evaluation techniques for each research experiment (Please refer to Chapter 4-6 for individual experiment details). The thesis project as a whole is controlled by applying Vaishnavi and Kuechler's (2007) model for design research.
Research Contributions	The major research contributions have been presented and justified in this chapter. They can be recapped as follows: - Bluetooth ping for proximity determination - Simple cloud to device messaging for improving cloud to device push messaging - Context-aware computation in the Google cloud - Exploitation of multi-dimensional context-aware information - Tailoring and adaptation of the user experience
Research Rigour	The research rigour is ensured by having artefact development and testing controlled by Scrum (Schwaber and Beedle, 2002) and thesis, as well as experiments, evaluation controlled by Hevner et al. (2004) guidelines.
Design as a Search Process	Considered fulfilled as the research aim is grounded in an awareness from the mobile computing business domain and further investigated, detailed and justified through the literature review. Additionally the thesis consists of three incremental research experiments building upon each other, adding to the knowledge domain leading to the conclusion answering the research goal and objectives.

Communication of Research	The research is communicated back to the research field by this thesis. Additionally the artefacts from Bluetooth ping and simple push to device messaging are released as open source projects. Last but not least, several publications have been published to the research community whilst fulfilling this work reporting on state of the art experiences and results from the field.

7.4 Limitations

The findings report interesting results from several dimensions, but nonetheless it is important to be aware of the limitations of the work. The Google cloud can be seen as a limitation off this research. The cloud provides several services but at the same time these services are out of reach, opening for server downtime vulnerability.

I acknowledge that since some evaluation questions concern personal preferences, the results could be somewhat different with a large population in the evaluations. I also acknowledge that the tests performed were on a wide, but still limited, selection of different smartphones. Variations in behaviour may be an issue when it comes to other vendor implementations. Solutions ranges from targeting one to four platforms and further research on different platforms and operating systems may yield aberrant results.

7.5 Further research

This research has shown the feasibility for tailoring and adaptation of the user experience based on multi-dimensional context-aware information. I believe further research should be conducted to widen the concept of context-awareness further. Work needs to be conducted to include even more sources for context-aware information. Near field communication (NFC), ad hoc services, social networks and crowd sourcing all represent interesting opportunities as sources of information. Creating a framework for abstracting the integration with all these services and the many to come should be considered. The implementation of such a framework should not be attempted before solutions for challenges such as cross-platform

integration, shared context state, uniform exchange of information between context-aware services are in place.

Context-awareness has undergone a tremendous transformation and development the last ten years. For the next ten years new innovations and technological innovations will continue to expand this field. The ultimate goal is still to create a simpler, more adapted, custom tailored, easy to use interface for the user – and this should be the all-encompassing goal for all further research pursuits.

Chapter 8

Bibliography

Abowd, G., Atkeson, C., Hong, J., Long, S., Kooper, R. & Pinkerton, M. 1997. Cyberguide: A mobile context‐aware tour guide. *Wireless Networks,* 3**,** 421-433.

Abowd, G. D., Ebling, M., Hung, G., Hui, L. & Gellersen, H. W. 2002. Context-aware computing. *Pervasive Computing, IEEE,* 99**,** 22-23.

Ahmed, S., Sharmin, M. & Ahamed, S. I. A Smart Meeting Room with Pervasive Computing Technologies. Sixth International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing 2005. IEEE Computer Society, 366-371.

Amazon. 2012. *Amazon Elastic Compute Cloud (Amazon EC2)* [Online]. Available: http://aws.amazon.com/ec2/ [Accessed March 13 2012].

Android. 2011. *ADT plugin for Eclipse* [Online]. Available: http://developer.android.com/sdk/eclipse-adt.html [Accessed March 8 2012].

Android. 2012a. *Android everywhere* [Online]. Available: http://www.android.com/developers/ [Accessed March 8 2012].

Android. 2012b. *What is Android? | Android Developers* [Online]. Available: http://developer.android.com/guide/basics/what-is-android.html [Accessed March 8th 2012].

Ankolekar, A., Szabo, G., Luon, Y., Huberman, B. A., Wilkinson, D. & Wu, F. 2009. Friendlee: a mobile application for your social life. *Proceedings of the 11th International Conference on Human-Computer Interaction with Mobile Devices and Services.* Bonn, Germany: ACM.

Apple Inc. 2012a. *Apple Developer Technologies - Overview* [Online]. Available: https://developer.apple.com/technologies/ [Accessed March 8 2012].

Apple Inc. 2012b. *Apple iPhone* [Online]. Available: http://www.apple.com/iphone/ [Accessed March 9 2012].

Apple Inc. 2012c. *Apple Mac* [Online]. Available: http://www.apple.com/mac/ [Accessed March 9 2012].

Apple Inc. 2012d. *Apple's App Store Downloads Top 25 Billion* [Online]. Available: http://www.apple.com/pr/library/2012/03/05Apples-App-Store-Downloads-Top-25-Billion.html [Accessed March 8 2012].

Armbrust, M., Fox, A., Griffith, R., Joseph, A. D., Katz, R., Konwinski, A., Lee, G., Patterson, D., Rabkin, A., Stoica, I. & Zaharia, M. 2010. A view of cloud computing. *Commun. ACM,* 53**,** 50-58.

Bbc News. 2011. *Cisco predicts internet device boom* [Online]. Available: http://www.bbc.co.uk/news/technology-13613536 [Accessed March 20 2012].

Beck, K. 1999. Embracing change with extreme programming. *Computer,* 32**,** 70-77.

Biegel, G. & Cahill, V. A framework for developing mobile, context-aware applications. Pervasive Computing and Communications, 2004. PerCom 2004. Proceedings of the Second IEEE Annual Conference on, 14-17 March 2004 2004. 361-365.

Bilandzic, M., Foth, M. & Luca, A. CityFlocks: designing social navigation for urban mobile informtaion systems. 7th ACM conference on Designing Interactive Systems, 2008. 174-183.

Binnig, C., Kossmann, D., Kraska, T. & Loesing, S. 2009. How is the weather tomorrow?: towards a benchmark for the cloud. *Proceedings of the Second International Workshop on Testing Database Systems.* Providence, Rhode Island: ACM.

Borenstein, N. & Blake, J. 2011. Cloud Computing Standards: Where's the Beef? *Internet Computing, IEEE,* 15**,** 74-78.

Bourcier, J., Escoffier, C. & Lalanda, P. Implementing Home- Control Applications on Service Platform. 4th IEEE Consumer Communications and Networking Conference, 2007. 925-929.

Brown, B. & Randell, R. 2004. Building a Context Sensitive Telephone: Some Hopes and Pitfalls for Context Sensitive Computing. *Computer Supported Cooperative Work (CSCW),* 13**,** 329-345.

Bush, J. O., Irvine, J. A. & Dunlop, J. Personal Assistant Agent and Content Manager for Ubiquitous Services. 3rd International Symposium on Wireless Communication Systems, 2006. 169-173.

Canalys. 2012. *Smart phones overtake client PCs in 2011* [Online]. Available: http://www.canalys.com/newsroom/smart-phones-overtake-client-pcs-2011 [Accessed March 15 2012].

Chen, G. & Kotz, D. 2000. A Survey of Context-Aware Mobile Computing Research. *Dartmouth Computer Science Technical Report TR2000-381.*

Chen, H., Perich, F., Chakraborty, D., Finin, T. & Joshi, A. Intelligent Agents Meet Semantic Web in a Smart Meeting Room. Third International Joint Conference on Autonomous Agents and Multiagent Systems 2004 New York, New York. IEEE Computer Society, 854-861.

Chen, W. & Hirschheim, R. 2004. A paradigmatic and methodological examination of information systems research from 1991 to 2001. *Information Systems Journal,* 14**,** 197-235.

Cheverst, K., Davies, N., Mitchell, K. & Efstratiou, C. 2001. Using Context as a Crystal Ball: Rewards and Pitfalls. *Personal Ubiquitous Comput.,* 5**,** 8-11.

Cheverst, K., Davies, N., Mitchell, K., Friday, A. & Efstratiou, C. 2000. Developing a context-aware electronic tourist guide: some issues and experiences. *Proceedings of the SIGCHI conference on Human factors in computing systems.* The Hague, The Netherlands: ACM.

Christensen, J. H. 2009. Using RESTful web-services and cloud computing to create next generation mobile applications. *Proceedings of the 24th ACM SIGPLAN conference companion on Object oriented programming systems languages and applications.* Orlando, Florida, USA: ACM.

Cross, N. 2001. Designerly Ways of Knowing: Design Discipline Versus Design Science. *Design Issues,* 17**,** 49-55.

Cutrell, E., Robbins, D., Dumais, S. & Sarin, R. Fast, Flexible Filtering with Phlat Personal Search and Organization Made Easy. Conference on Human Computer Factors in Computing Systems, 2006. 261-270.

Dai, P. & Xu, G. Context-aware computing for assistive meeting system. 1st international conference on PErvasive Technologies Related to Assistive Environments, 2008 Athens, Greece. ACM, 1-7

Davies, N., Cheverst, K., Mitchell, K. & Efrat, A. 2001. Using and determining location in a context-sensitive tour guide. *Computer,* 34**,** 35-41.

Dey, A. 2001. Understanding and Using Context. *Journal of Personal and Ubiquitous Computing,* 5**,** 4-7.

Dey, A. & Abowd, G. 2000. CybreMinder: A Context-Aware System for Supporting Reminders

Handheld and Ubiquitous Computing. *In:* Thomas, P. & Gellersen, H.-W. (eds.). Springer Berlin / Heidelberg.

Dey, A., Abowd, G., Salber, D. & Futakawa, M. An Architecture to Support Context-Aware Applications. 12th Annual ACM Symposium on User Interface Software and Technology, 1999.

Dey, A. & Abowd, G. D. Towards a Better Understanding of Context and Context-Awareness. 1st international symposium on Handheld and Ubiquitous Computing, 1999.

Dey, A., Salber, D. & Abowd, G. 2001. A conceptual framework and a toolkit for supporting the rapid prototyping of context-aware applications. *Human-Computer Interaction,* 16**,** 97-166.

Dimarzio, J. 2008. *Android: A Programmer's Guide,* Chicago, United States of America, McGraw-Hill.

Dourish, P. 2004. What we talk about when we talk about context. *Personal Ubiquitous Comput.,* 8**,** 19-30.

Dubochet, G. Computer Code as a Medium for Human Communication: Are Programming Languages Improving? 21st Annual Workshop of the Psychology of Programming Interest Group, 2009.

Echtibi, A., Zemerly, M. J. & Berri, J. 2009. Murshid: a mobile tourist companion. *Proceedings of the 1st International Workshop on Context-Aware Middleware and Services: affiliated with the 4th International Conference on Communication System Software and Middleware (COMSWARE 2009).* Dublin, Ireland: ACM.

Economist 2011. Ubiquitous computing: Up close. *The Economist.*

Edwards, W. K. 2005. Putting computing in context: An infrastructure to support extensible context-enhanced collaborative applications. *ACM Transactions on Computer-Human Interaction (TOCHI),* 12**,** 446-474.

Fahy, P. & Siobhan, C. CASS-middleware for mobile context-aware applications. Workshop on Context Awareness MobiSys, 2004. 304-308.

Ferris, B., Watkins, K. & Borning, A. 2010. Location-Aware Tools for Improving Public Transit Usability.

Fred.E, F. 1964. A Contingency Model of Leadership Effectiveness. *In:* Leonard, B. (ed.) *Advances in Experimental Social Psychology.* Academic Press.

Gartner Group. 2011a. *Gartner Identifies the Top 10 Strategic Technologies for 2012* [Online]. Available: http://www.gartner.com/it/page.jsp?id=1826214 [Accessed March 20 2012].

Gartner Group. 2011b. *Gartner Says Android to Command Nearly Half of Worldwide Smartphone Operating System Market by Year-End 2012*

[Online]. Available: http://www.gartner.com/it/page.jsp?id=1622614 [Accessed December 1 2011].

Gartner Group. 2011c. *Gartner Says Sales of Mobile Devices Grew 5.6 Percent in Third Quarter of 2011* [Online]. Available: http://www.gartner.com/it/page.jsp?id=1848514 [Accessed February 14 2012].

Gartner Group. 2012a. *Gartner Says Apple Became the Top Semiconductor Customer in 2011* [Online]. Available: http://www.gartner.com/it/page.jsp?id=1902414 [Accessed March 20 2012].

Gartner Group. 2012b. *Gartner Says Worldwide Smartphone Sales Soared in Fourth Quarter of 2011 With 47 Percent Growth* [Online]. Available: http://www.gartner.com/it/page.jsp?id=1924314 [Accessed March 20 2012].

Gkonis, P., Patrikakis, C., Anadiotis, A.-C., Kaklamani, D., Andrade, M., Detti, A., Tropea, G. & Melazzi, N. A content-centric, publish-subscribe architecture delivering mobile context-aware health services. IEEE Future Network & Mobile Summit (FutureNetw), 2011. 1-9.

Göker, A., Watt, S., Myrhaug, H. I., Whitehead, N., Yakici, M., Bierig, R., Nuti, S. K. & Cumming, H. 2004. An ambient, personalised, and context-sensitive information system for mobile users. *Proceedings of the 2nd European Union symposium on Ambient intelligence.* Eindhoven, Netherlands: ACM.

Google. 2011. *Protocol Buffers - Google's data interchange format* [Online]. Available: http://code.google.com/p/protobuf/ [Accessed March 10 2012].

Google. 2012. *What Is Google App Engine?* [Online]. Available: http://code.google.com/appengine/docs/whatisgoogleappengine.html [Accessed March 8 2012].

Google. n.d.-a. *Developer Guide* [Online]. Available: http://code.google.com/apis/protocolbuffers/docs/overview.html [Accessed March 10 2012].

Google. n.d.-b. *Google App Engine* [Online]. Available: http://code.google.com/appengine/ [Accessed March 13 2012].

Goyal, S., Choudhary, V. & Khan, I. 2012. Cloud Computing On Mobile Multimedia Application. *International Journal of Electronics Communication and Computer Engineering,* 3.

Gregg, D. G., Kulkarni, U. R., Vinz\, A. S. & \#233 2001. Understanding the Philosophical Underpinnings of Software Engineering Research in Information Systems. *Information Systems Frontiers,* 3**,** 169-183.

Grix, J. 2004. *The foundations of research*, Palgrave Macmillan.

Gsmarena. 2012. *GSMArena.com* [Online]. Available: http://www.gsmarena.com/ [Accessed April 4 2012].

Guan, L., Ke, X., Song, M. & Song, J. A Survey of Research on Mobile Cloud Computing. IEEE/ACIS 10th International Conference on Computer and Information Science, 16-18 May 2011 2011. 387-392.

Häkkilä, J. & Mäntyjärvi, J. 2004. User Experiences on Combining Location Sensitive Mobile Phone Applications and Multimedia Messaging. *Conference of Mobile and Ubiquitous Multimedia.*

Hall, S. P. & Anderson, E. 2009. Operating systems for mobile computing. *J. Comput. Small Coll.,* 25**,** 64-71.

Henricksen, K. & Indulska, J. 2005. Personalising Context-Aware Applications
On the Move to Meaningful Internet Systems 2005: OTM 2005 Workshops. *In:* Meersman, R., Tari, Z. & Herrero, P. (eds.). Springer Berlin / Heidelberg.

Hertzog, P. & Torrens, M. Context-aware mobile Assistants for Optimal Interaction: a Prototype for Supporting the Business Traveler. 9th international conference on Intelligent user interfaces, 2004. 256-258.

Hevner, A. & Chatterjee, S. 2010. *Design Research in Information Systems: Theory and Practice*, Springer.

Hevner, A. R., March, S. T., Park, J. & Ram, S. 2004. Design Science in Information Systems Research. *MIS Quarterly,* 28**,** 75-105.

Hofer, T., Schwinger, W., Pichler, M., Leonhartsberger, G., Altmann, J. & Retschitzegger, W. Context-awareness on mobile devices - the hydrogen approach. System Sciences, 2003. Proceedings of the 36th Annual Hawaii International Conference on, 6-9 Jan. 2003 2003. 10 pp.

Hong, J.-Y., Suh, E.-H. & Kim, S.-J. 2009. Context-aware systems: A literature review and classification. *Expert Systems with Applications,* 36**,** 8509-8522.

Htc. 2012. *HTC Europe* [Online]. Available: http://www.htc.com/europe/ [Accessed March 9 2012].

Huang, D. 2011. Mobile Cloud Computing. *IEEE Communications Society MMTC*

Izettle. 2012. *iZettle* [Online]. Available: http://www.izettle.com/ [Accessed March 20 2012].

Johns, R. 2010. *Likert Items and Scales* [Online]. Available: http://surveynet.ac.uk/sqb/datacollection/likertfactsheet.pdf [Accessed March 8 2012].

Kaasinen, E. 2003. User needs for location-aware mobile services. *Personal and Ubiquitous Computing,* 7**,** 70-79.

Kabassi, K. 2010. Review: Personalizing recommendations for tourists. *Journal Telematics and Informatics,* 27**,** 51-66.

Kang, S., Lee, J., Jang, H., Lee, H., Lee, Y., Park, S., Park, T. & Song, J. 2008. SeeMon: scalable and energy-efficient context monitoring framework for sensor-rich mobile environments. *Proceedings of the 6th international conference on Mobile systems, applications, and services.* Breckenridge, CO, USA: ACM.

Kaptein, M. C., Nass, C. & Markopoulos, P. 2010. Powerful and consistent analysis of likert-type ratingscales. *Proceedings of the 28th international conference on Human factors in computing systems.* Atlanta, Georgia, USA: ACM.

Kenteris, M., Gavalas, M. D. & Economou, D. Developing Tourist Guide Applications for Mobile Devices using the J2ME Platform. Mobile Computing and Wireless Communication International Conference, 2006. MCWC 2006. Proceedings of the First, 17-20 Sept. 2006 2006. 178-183.

Khan, A. & Ahirwar, K. K. 2011. Mobile Cloud Computing as a Future of Mobile Multimedia Database. *International Journal of Computer Science and Communication,* 2**,** 219-221.

Kjeldskov, J. & Paay, J. 2006. Indexical interaction design for context-aware mobile computer systems. *Proceedings of the 18th Australia conference on Computer-Human Interaction: Design: Activities, Artefacts and Environments.* Sydney, Australia: ACM.

Klein, A., Mannweiler, C., Schneider, J. & Schotten, H. Access Schemes for Mobile Cloud Computing. Eleventh International Conference on Mobile Data Management, 2010.

Koller, A., Foster, G. & Wright, M. 2008. Java Micro Edition and Adobe Flash Lite for arcade-style mobile phone game development: a comparative study. *Proceedings of the 2008 annual research conference of the South African*

Institute of Computer Scientists and Information Technologists on IT research in developing countries: riding the wave of technology. Wilderness, South Africa: ACM.

Korpipaa, P., Mantyjarvi, J., Kela, J., Keranen, H. & Malm, E. J. 2003. Managing context information in mobile devices. *Pervasive Computing, IEEE,* 2**,** 42-51.

Kuechler, B. & Vaishnavi, V. 2008. On theory development in design science research: anatomy of a research project. *European Journal of Information Systems,* 17**,** 489-504.

Kurkovsky, S. & Harihar, K. 2006. Using ubiquitous computing in interactive mobile marketing. *Personal and Ubiquitous Computing,* 10**,** 227-240.

Larman, C. 2002. *Applying UML and patterns: an introduction to object-oriented analysis and design and the unified process*, Prentice Hall PTR.

Leary, M. R. 1995. *Introduction to behavioral research methods*, Brooks/Cole.

Leiba, B. 2009. Having One's Head in the Cloud. *Internet Computing, IEEE,* 13**,** 4-6.

Lincoln, D., Endler, M., Barbosa, S. & Filho, J. V. Middleware Support for Context-Aware Mobile Applications with Adaptive Multimodal User Interfaces. Ubi-Media Computing 3-4 July 2011 2011. 106-111.

Loke, S. 2006. *Context-Aware Pervasive Systems: Architectures for a New Breed of Applications,* Boston, Ma, USA, Auerbach Publications.

Long, S., Kooper, R., Abowd, G. D. & Atkeson, C. G. 1996. Rapid prototyping of mobile context-aware applications: the Cyberguide case study. *Proceedings of the 2nd annual international conference on Mobile computing and networking.* Rye, New York, United States: ACM.

Lovett, T. & O'neill, E. Mobile Context-Awareness: Capabilities, Challenges and Applications. UBICOMP, 2010. 539-540.

Ludford, P., Rankowski, D., Reily, K., Wilms, K. & Terveen, L. Because I carry my cell phone anyway: functional location-based reminder applications. Conference on Human Factors in Computing Systems, 2006. 889-898.

March, S. T. & Smith, G. F. 1995. Design and natural science research on information technology. *Decis. Support Syst.,* 15**,** 251-266.

Mcmurtrey, K. Defining the Out-of-the-Box experience: A case study. Annual conference Society for Technical Communication, 2001.

Mei, L., Chan, W. K. & Tse, T. H. A Tale of Clouds: Paradigm Comparisons and Some Thoughts on Research Issues. Proceedings of the 2008 IEEE Asia-Pacific Services Computing Conference, 2008. IEEE Computer Society, 464-469.

Mei, L., Zhang, Z. & Chan, W. K. 2009. More Tales of Clouds: Software Engineering Research Issues from the Cloud Application Perspective. *Proceedings of the 2009 33rd Annual IEEE International Computer Software and Applications Conference - Volume 01.* IEEE Computer Society.

Mell, P. & Grance, T. 2011. The NIST Definition of Cloud Computing.

Michahelles, F. & Samulowitz, M. 2002. Smart CAPs for Smart Its – Context Detection for Mobile Users. *Journal Personal and Ubiquitous Computing,* 6.

Microsoft. 2005. *About Windows CE .NET* [Online]. Available: http://msdn.microsoft.com/en-us/library/ms920098.aspx [Accessed March 8 2012].

Microsoft. 2012a. *Pocket Outlook Object Model (POOM) API* [Online]. Available: http://msdn.microsoft.com/en-us/library/ms897378.aspx [Accessed March 9 2012].

Microsoft. 2012b. *Windows Phone* [Online]. Available: http://www.microsoft.com/windowsphone/en-us/default.aspx [Accessed March 8 2012].

Mokbel, M. F. & Levandoski, J. J. 2009. Toward context and preference-aware location-based services. *Proceedings of the Eighth ACM International Workshop on Data Engineering for Wireless and Mobile Access.* Providence, Rhode Island: ACM.

Mowbray, M. & Pearson, S. 2009. A client-based privacy manager for cloud computing. *Proceedings of the Fourth International ICST Conference on COMmunication System softWAre and middlewaRE.* Dublin, Ireland: ACM.

Munoz, M. A., Rodriguez, M., Favela, J., Martinez-Garcia, A. I. & Gonzalez, V. M. 2003. Context-aware mobile communication in hospitals. *Computer,* 36**,** 38-46.

Muralidharan, K. & Gupta, P. Nimbus: a task aware/context aware mobile computing platform. Software Technologies for Future Embedded and Ubiquitous Systems 27-28 April 2006 2006. 6 pp.

Myers, M. D. & Avison, D. E. 2002. *Qualitative Research in Information Systems*, Sage Publications.

Niehaves, B. 2007. On epistemological pluralism in design science. *Scandinavian Journal of Information Systems,* 19**,** 93-104.

Nijholt, A., Rienks, R., Zwiers, J. & Reidsma, D. 2006. Online and off-line visualization of meeting information and meeting support. *The Visual Computer* 22**,** 965-976

Nokia. 2012a. *Nokia Developer - Devices Overview* [Online]. Available: http://www.developer.nokia.com/Devices/ [Accessed March 9 2012].

Nokia. 2012b. *Symbian platform* [Online]. Available: http://www.developer.nokia.com/Devices/Symbian/ [Accessed April 5 2012].

Open Handset Alliance. 2007. *Industry Leaders Announce Open Platform for Mobile Devices* [Online]. Available: http://www.openhandsetalliance.com/press_110507.html [Accessed March 20 2012].

Oracle. n.d. *Java for Mobile Devices* [Online]. Available: http://www.oracle.com/technetwork/java/javame/index.html [Accessed March 6 2012].

Orlikowski, W. & Baroudi, J. 1991. Studying Information Technology in Organizations: Research Approaches and Assumptions. *Information Systems Research,* 2.

Oslo Kommune. 2004. *Bydelskart for Oslo* [Online]. Available: http://www.oslo.kommune.no/getfile.php/oslokommune(OSLO)/Internett(OSLO)/Dokumenter/dokument/kart2.pdf [Accessed May 20th 2008].

Oulasvirta, A., Raento, M. & Tiitta, S. 2005. ContextContacts: re-designing SmartPhone's contact book to support mobile awareness and collaboration. *Proceedings of the 7th international conference on Human computer interaction with mobile devices \& services.* Salzburg, Austria: ACM.

Pallant, J. 2011. *SPSS survival manual : a step by step guide to data analysis using SPSS,* Crows Nest, N.S.W., Allen & Unwin.

Parviainen, M., Pirinen, T. & Pertilä, P. 2006. A Speaker Localization System for Lecture Room Environment. *Machine Learning for Multimodal Interaction***,** 225-235.

Pascoe, J. Adding generic contextual capabilities to wearable computers. Wearable Computers, 1998. Digest of Papers. Second International Symposium on, 19-20 Oct 1998 1998. 92-99.

Peffers, K., Tuunanen, T., Rothenberger, M. & Chatterjee, S. 2007. A Design Science Research Methodology for Information Systems Research. *Journal of Management Information Systems,* 24**,** 45-77.

Perttunen, M., Riekki, J., Koskinen, K. & Tahti, M. Experiments on mobile context-aware instant messaging. Collaborative Technologies and Systems, 2005. Proceedings of the 2005 International Symposium on, 20-20 May 2005 2005. 305-312.

Philipose, M., Fishkin, K. P., Perkowitz, M., Patterson, D. J., Fox, D., Kautz, H. & Hahnel, D. 2004. Inferring activities from interactions with objects. *Pervasive Computing, IEEE,* 3**,** 50-57.

Prekop, P. & Burnett, M. 2003. Activities, Context and Ubiquitous Computing. *Journal of Computer Communications, Special Issue on Ubiquitous Computing,* 26**,** 1168-1176.

Qureshi, S. S., Ahmad, T., Rafique, K. & Shuja Ul, I. Mobile cloud computing as future for mobile applications - Implementation methods and challenging issues. Cloud Computing and Intelligence Systems (CCIS), 2011 IEEE International Conference on, 15-17 Sept. 2011 2011. 467-471.

Rodden, T., Chervest, K., Davies, N. & Dix, A. Exploiting Context in {HCI} Design for Mobile Systems. in Workshop on Human Computer Interaction with Mobile Devices, 1998.

Salesforce. 2012. *CRM - The Enterprise Cloud* [Online]. Available: http://www.salesforce.com/eu/?ir=1 [Accessed May 28 2012].

Samsung. 2012. *Galaxy Tab Android Tablet* [Online]. Available: http://www.samsung.com/us/mobile/galaxy-tab [Accessed March 9 2012].

Sarker, S. & Wells, J. D. 2003. Understanding mobile handheld device use and adoption. *Commun. ACM,* 46**,** 35-40.

Satyanarayanan, M. 1996. Fundamental challenges in mobile computing. *Proceedings of the fifteenth annual ACM symposium on Principles of distributed computing.* Philadelphia, Pennsylvania, United States: ACM.

Satyanarayanan, M. 2011. Mobile computing: the next decade. *SIGMOBILE Mob. Comput. Commun. Rev.,* 15**,** 2-10.

Schilit, B., Adams, N. & Want, R. Context-Aware Computing Applications. Proceedings of the 1994 First Workshop on Mobile Computing Systems and Applications, 1994. IEEE Computer Society, 85-90.

Schmidt, A. 2000. Implicit human computer interaction through context. *Personal Technologies,* 4**,** 191-199.

Schmidt, A. 2010. Ubiquitous Computing: Are We There Yet? *Computer,* 43**,** 95-97.

Schmidt, A., Aidoo, K., Takaluoma, A., Tuomela, U., Van Laerhoven, K. & Van De Velde, W. 1999a. Advanced Interaction in Context

Handheld and Ubiquitous Computing. *In:* Gellersen, H.-W. (ed.). Springer Berlin / Heidelberg.

Schmidt, A., Beigl, M. & Gellersen, H.-W. 1999b. There is more to context than location. *Computers & Graphics,* 23**,** 893-901.

Schmidt, A., Stuhr, T. & Gellersen, H. 2001. Context-phonebook – Extending Mobile Phone Applications with Context.

Schwaber, K. & Beedle, M. 2002. *Agile Software Development with Scrum*, Prentice Hall.

Schwinger, W., Grün, C., Pröll, B. & Retschitzegger, W. 2005. Context-Awareness in Mobile Tourist Guides. *Handbook of Research on Mobile Multimedia.* IGI Global.

Scrum Alliance. n.d. *What is Scrum* [Online]. Available: http://www.scrumalliance.org/pages/what_is_scrum [Accessed Feb 18 2012].

Shiels, M. 2010. *Mobile application sales to reach $17.5bn by 2012* [Online]. Available: http://news.bbc.co.uk/2/hi/technology/8571210.stm [Accessed February 10 2012].

Shrestha, A. MobileSOA Framework for Context-Aware Mobile Applications. Mobile Data Management (MDM), 2010 Eleventh International Conference on, 23-26 May 2010 2010. 297-298.

Siewiorek, D., Smailagic, A., Furukawa, J., Krause, A., Moraveji, N., Reiger, K., Shaffer, J. & Wong, F. L. SenSay: A Context-Aware Mobile Phone. Seventh IEEE International Symposium on Wearable Computers, 2003.

Simcock, T., Hillenbrand, S. P. & Thomas, B. H. 2003. Developing a location based tourist guide application. *Proceedings of the Australasian information security workshop conference on ACSW frontiers 2003 - Volume 21.* Adelaide, Australia: Australian Computer Society, Inc.

Simoens, P., De Turck, F., Dhoedt, B. & Demeester, P. 2011. Remote Display Solutions for Mobile Cloud Computing. *IEEE Computer,* 44**,** 46-53.

Simon, H. 1947. *Administrative Behavior*, Free Press.

Spriestersbach, A., Vogler, H., Lehmann, F. & Ziegert, T. 2001. Integrating context information into enterprise applications for the mobile workforce - a case study. *Proceedings of the 1st international workshop on Mobile commerce.* Rome, Italy: ACM.

Square. 2012. *Accept Credit Cards with Your iPhone, Android or iPad - Square* [Online]. Available: https://squareup.com/ [Accessed March 20 2012].

Strobbe, M., Van Laere, O., Ongenae, F., Dauwe, S., Dhoedt, B., De Turck, F., Demeester, P. & Luyten, K. 2011. Integrating Location and Context Information for Novel Personalised Applications. *Pervasive Computing, IEEE,* PP**,** 1-1.

Tamminen, S., Oulasvirta, A., Toiskallio, K. & Kankainen, A. 2004. Understanding mobile contexts. *Personal Ubiquitous Comput.,* 8**,** 135-143.

Vaishnavi & Kuechler. 2004. *Design Science Research in Information Systems* [Online]. Available: http://desrist.org/desrist [Accessed 27 October 2011].

Vaishnavi, V. & Kuechler, W. 2007. *Design science research methods and patterns: innovating information and communication technology*, Auerbach Publications.

Vaus, D. a. D. 2001. *Research design in social research*, SAGE.

Venable, J. The Role of Theory and Theorising in Design Science Research. Proceedings of the First International Conference on Design Science in Information Systems and Technology (DESRIST), 2006.

Visa. 2012. *The Visa Mobile Platform — the Future of Payments* [Online]. Available: http://visa.ca/merchant/products-and-services/mobile-platform.jsp [Accessed March 20 2012].

Walsham, G. 1995. Interpretive case studies in IS research: nature and method. *European Journal Information Systems,* 4**,** 197-235.

Want, R., Schilit, B., Adams, N., Gold, R., Petersen, K., Goldberg, D., Ellis, J. & Weiser, M. 1995. The ParcTab Ubiquitous Computing Experiment. Xerox.

Wei, E. & Chan, A. 2007. Towards Context-Awareness in Ubiquitous Computing

Embedded and Ubiquitous Computing. *In:* Kuo, T.-W., Sha, E., Guo, M., Yang, L. & Shao, Z. (eds.). Springer Berlin / Heidelberg.

Weiser, M. 1991. The Computer for the 21st Century. *Scientific American,* 265**,** 99-104.

Weiser, M. 1993. Some computer science issues in ubiquitous computing. *Commun. ACM,* 36**,** 75-84.

Weiser, M. 1994. The world is not a desktop. *interactions,* 1**,** 7-8.

Winter, R. 2008. Design science research in Europe. *European Journal of Information Systems,* 17**,** 470-475.

Woerndl, W., Brocco, M. & Eigner, R. 2009. Context-Aware Recommender Systems in Mobile Scenarios. *International Journal of Information Technology and Web Engineering,* 4**,** 67-85.

Wojciechowski, M. 2009. End User Context Modeling in Ambient Assisted Living. *International Journal of Advanced Pervasive and Ubiquitous Computing,* 1**,** 61-80.

Zhou, M., Houck, K., Pan, S., Shaw, J., Aggarwal, V. & Wen, Z. 2006. Enabling Context-Sensitive Information Seeking. *Conference of Intelligent User Interfaces.*

APPENDICES

Appendix A Questionnaire for Experiment 1

All participants completed this questionnaire immediately after completing the practical test with the PIM device. The questionnaire consists of three categories with five questions in each category. All questions have six possible answers as well as an opportunity to comment on the answer.

Evaluation after test of PIM application

In every question rate your answer according to:

Strongly Disagree | Disagree | Mildly Disagree | Mildly Agree | Agree | Strongly Agree

Initial Use:

Question 1:

It was easy to set up an appointment

SD D MD MA A SA

Comments:

Question 2:

It was difficult to find the "*meeting"* context category

SD D MD MA A SA

Comments:

Question 3:

It was difficult to navigate through the application

SD D MD MA A SA

Comments:

Question 4:

Finding the GPS controls and starting the GPS was easy

SD D MD MA A SA

Comments:

Question 5:

Getting a position with the GPS was difficult

SD D MD MA A SA

Comments:

PIM in action:

Question 6:

The information provided by the reminder system correctly matched my current location

SD D MD MA A SA

Comments:

Question 7:

The information provided when I was *"Sightseeing in old town"* was incorrect.

SD D MD MA A SA

Comments:

Question 8:

The summary of blocked events I received after appointment *"DNB Nor Solli plass"* was useful

SD D MD MA A SA

Comments:

Question 9:

The system provides duplicated information

SD D MD MA A SA

Comments:

Question 10:

I liked the fact that the application is integrated with Outlook

SD D MD MA A SA

Comments:

Overall statements:

Question 11:

I would have liked to receive info about alternative transportation in "*train"* context

SD D MD MA A SA

Comments:

Question 12:

The reminder system is useful

SD D MD MA A SA

Comments:

Question 13:

The application was slow

SD D MD MA A SA

Comments:

Question 14:

The device got warm as I performed my schedule

SD D MD MA A SA

Comments:

Question 15:

I would use this application in my daily life

SD D MD MA A SA

Comments:

Question 16:

Do you have any overall comments or suggestions for improvements?

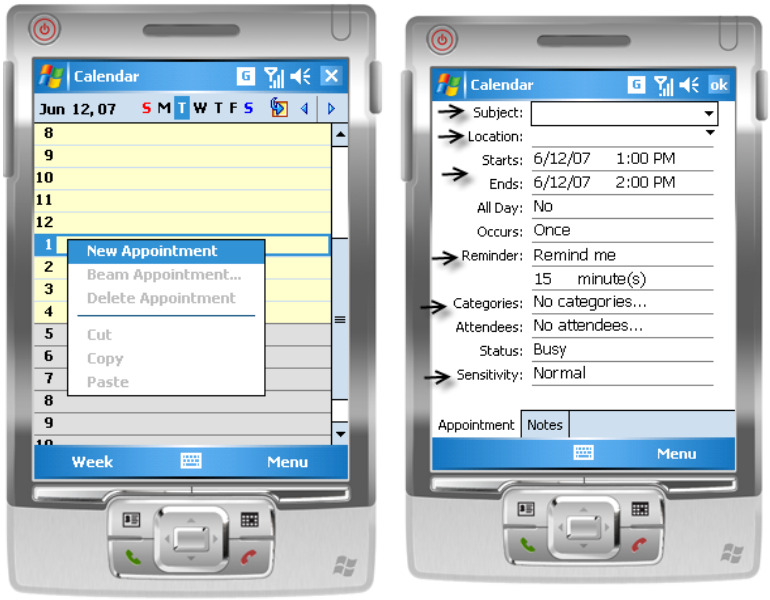
Comments:

Thank you for participating!

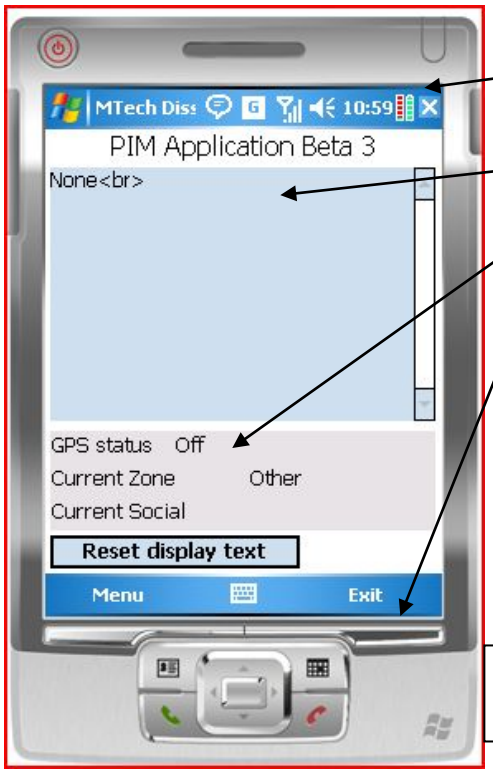
Appendix B

User Guide for Experiment 1

This mini user guide was handed out to the users testing the application.



When calendar is open, tap and hold to add appointment. Then fill in the information marked with an arrow according to the information described in the Pocket Outlook Schedule document.



PIM application running

- Tap X to minimize application

- Main area for display of information

- Status information

- Exit will close the application

- Messages generated from the system will occur as notification messages.

General phone information:
Phone number: [redacted]
PIN code: 1111
PUK code: 59448395

Restart phone and application by pressing stylus tip into the soft reset opening

Appendix C

Questionnaire for Experiment 2

All participants completed this questionnaire immediately after completing the practical test. The questionnaire consists of seven categories with up to five questions in each category. All questions have four possible answers as well as an opportunity to comment on the answer.

In every question rate your answer according to:

Strongly Disagree | Disagree | Agree | Strongly Agree

Theme / question number	**Statement**
1. The user interface	
	It is easy to see the available functions.
	The features of the application are hard to use.
2. Performance	
	Browsing back and forward in the current meeting meeting-notes is quick.
	The presentation notes displayed on the mobile device was synchronised with the audio presentation/presenter's narrative
3. User features	
	I was not able to register as a participant in the web application
	I was able to look at next and previous notes during the presentation

4. Context-aware information	
	When the presenter started the meeting my mobile device was found and I began receiving meeting notes once in presentation mode
	The close integration with Google calendar is an inconvenience, I am not able to use the system without changing my existing or creating a new e-mail account at Google
	I do not like sharing my personal information (like my name and e-mail address) to a service that stores the information in the cloud.
	Storing meeting information in the Google cloud and combining this with personal information on the device is a useful feature.
5. Application features	
	I find the ability to navigate notes during the presentation useful
	I do not find the "meeting note" history functionality useful.
	I do not find the presenter/participant model useful for a meeting environment
	I would like to receive more content during the presentation (images, videos, files, web-pages etc)
6. Overall impression	
	I find the context-aware meeting room application useful
7. Open comment question	

Appendix D
User Instruction Sheet for the Context-Aware Meeting Room

Before meeting

1. Open the browser on the following URL:
   http://context-aware-meeting-room.appspot.com/meeting
2. Register your mobile device with the provided Bluetooth address and your personal email address.

During meeting

3. When entering the meeting room, verify that the welcome message is received.
4. During the presentation, verify that meeting notes are received when the presenter changes slide.
5. Verify that the meeting note received is in synch with the current slide.
6. Move to a previously received meeting note.
7. Move to most recent meeting note.
8. Disconnect the mobile client from the current meeting.

After meeting

9. After the meeting has ended, use the history feature to view notes from a previously held meeting.
10. Navigate back and forth in the notes from this meeting.

Appendix E

Questionnaire for Experiment 3

All participants completed this questionnaire immediately after completing the practical test. The questionnaire consists of seven categories with up to five questions in each category. All questions have four possible answers as well as an opportunity to comment on the answer.

Evaluation after test of the application

In every question rate your answer according to:

Strongly Disagree | Disagree | Agree | Strongly Agree

Theme / question number	**Statement**
Statements regarding user interface	
	It is easy to see the available functions
	The features of the application are hard to use
	The adaptability of the application is a feature I approve
Statements regarding sensor integration	
	The background color in the application changes when the lighting in the room changes
	When moving around, a simplified user interface is not presented
	I find sensor integration annoying and would disable it on my device
Statements regarding the web application	
	I was able to register my device application configuration in the web application

	I was not able to store and push my configuration to my mobile device from the web page
	I would like to configure my phone from a cloud service on a daily basis, (webpage user config and Google services like mail/calendar/contacts)
Statements regarding context-awareness	
	The close integration with Google services is an inconvenience, I am not able to use the system without changing my existing or creating a new e-mail account at Google
	Calendar appointments displayed matched my current user context
	The contacts displayed did not match my current user context
	I would like to see integration with other online services such as online editing tools (for example Google Docs) and user messaging applications (like Twitter and Google Buzz)
Statements regarding cloud computing	
	I do not mind Cloud server downtime
	I do not like sharing my personal information (like my name and e-mail address) to a service that stores the information in the cloud
	Storing data in the Google Cloud and combining this with personal information on the device is a useful feature.
	I find the cloud-to-device application useful.
Overall satisfaction	
	I find the cloud-to-device application useful
Open comment question	

Appendix F

Instruction Sheet for Experiment 3

Username: brunel.nith

Password: ac.uk.no

Experiment walkthrough:

1. Start Android application, you might need to enter username/password and give the application access to the Google account the first time.

2. Select calendar tab
    1. Pay attention to the type of the first appointment

3. Select contacts tab.
    1. Make sure the contacts are filtered based on the *type* tag on the next meeting. If work is the type of the next meeting only work related contacts should be shown.

4. Move to the sensor tab on the Android device.
    1. Select the different sensor options and look at how the values changes when you move/shake/expose to light etc.

5. On your computer start a browser and log in to webpage: http://home-screen-cloud.appspot.com
    1. Select the application you want displayed on the home-screen on the Android device.
    2. Press the "Save configuration" button. The configuration is sent as a push message directly to your phone when this button is pressed.

6. Make sure the home-screen on the Android device (Apps tab) is updated and the icons you selected on the webpage are shown.

7. Press one of the icons to launch the application and get back into the application afterwards

8. On the apps-screen of the Android client, try to expose the phone to more/less light.

9. Make sure the background changes color based on the different light levels.

10. Still on the apps-screen, pick up the phone and shake it.

11. The UI should now change to a simple layout. This is meant to be shown to users on the move (for example running/walking) where they might want to use a simpler and more concise layout. After 15 seconds of keeping the device still, the usual layout is displayed

12. Log in to Google calendar in your browser: http://calendar.google.com

13. Experiment with the calendar and contacts integration by moving existing appointments, and make sure the contacts shown are updated.

14. Please answer the questionnaire: http://www.surveymonkey.com/s/androidHomeScreen

Appendix G

Source Code Excerpt Experiment 1

For detailed source code listing, please visit:

https://github.com/tormorten/brunelPhD/tree/master/experiment_1

Excerpt from Windows Mobile Client

Excerpt from Gps.cs

```
public GpsPosition GetPosition (TimeSpan maxAge)
{
  GpsPosition gpsPosition = null;
  if (Opened) {
    IntPtr ptr = Utils.LocalAlloc (Marshal.SizeOf (typeof(GpsPosition)));
    gpsPosition = new GpsPosition ();
    gpsPosition.dwVersion = 1;
    gpsPosition.dwSize = Marshal.SizeOf (typeof(GpsPosition));
    Marshal.StructureToPtr (gpsPosition, ptr, false);
    int result = GPSGetPosition (gpsHandle, ptr, 500000, 0);
    if (result == 0) {
      gpsPosition = (GpsPosition)Marshal.PtrToStructure (ptr, typeof(GpsPosition));
      if (maxAge != TimeSpan.Zero) {
        if (!gpsPosition.TimeValid || DateTime.Now -maxAge > gpsPosition.Time) {
          gpsPosition = null;
        }
      }
    }
    Utils.LocalFree (ptr);
  }
  return gpsPosition;
}
```

```
    //PInvokes to gpsapi.dll
    [DllImport("gpsapi.dll")]
    static extern IntPtr GPSOpenDevice (IntPtr hNewLocationData,
IntPtr hDeviceStateChange, string szDeviceName, int dwFlags);

    [DllImport("gpsapi.dll")]
    static extern int GPSCloseDevice (IntPtr hGPSDevice);

    [DllImport("gpsapi.dll")]
    static extern int GPSGetPosition (IntPtr hGPSDevice, IntPtr
pGPSPosition, int dwMaximumAge, int dwFlags);

    [DllImport("gpsapi.dll")]
    static extern int GPSGetDeviceState (IntPtr pGPSDevice);
    // PInvokes to coredll.dll
    [DllImport("coredll.dll")]
    static extern IntPtr CreateEvent (IntPtr lpEventAttributes, int
bManualReset, int bInitialState, StringBuilder lpName);

private void WaitForGpsEvents ()
{
  lock (this) {
    bool listening = true;
    IntPtr handles = Utils.LocalAlloc (12);
    Marshal.WriteInt32 (handles, 0, stopHandle.ToInt32 ());
    Marshal.WriteInt32 (handles, 4, deviceStateChangedHandle.ToInt32 ());
    Marshal.WriteInt32 (handles, 8, newLocationHandle.ToInt32 ());
    while (listening) {
      int obj = WaitForMultipleObjects (3, handles, 0, -1);
      if (obj != waitFailed) {
        switch (obj) {
        case 0:
                          // stop signalled triggered
          listening = false;
          break;
        case 1:
                           // device state has changed
          if (deviceStateChanged != null) {
            deviceStateChanged (this, new DeviceStateChangedEventArgs (GetDeviceState ()));
          }
          break;
        case 2:
                           // location has changed
          if (locationChanged != null) {
            if (zones.Moved (GetPosition ())) {

              zones.setZone (this.GetPosition ().dblLatitude, this.GetPosition ().dblLongitude);
              Business.Logger.
              LogEvent (this.GetPosition ().dblLatitude + " : " +
                        this.GetPosition ().dblLongitude + " : " + zones.ZoneName);
              locationChanged (this, new LocationChangedEventArgs (GetPosition ()));
            }
          }
          break;
        }
      }
    }
    Utils.LocalFree (handles);
    gpsEventThread = null;
  }
}
```

Excerpt from AppointmentItem.cs

```
public ArrayList getAllNewAppointments ()
{
  ArrayList alListe = new ArrayList ();
  if (Osession != null) {
    AppointmentCollection appointments = Osession.Appointments.Items;
    foreach (Appointment ap in appointments) {
      //34-36 ok if "Locale" is properly set in device, preferably 24h
      if (ap.Start > System.DateTime.Now) {
        alListe.Add (new Business.AppointmentVO (
                        ap.ItemId.ToString (), ap.Subject,
          ap.Categories, ap.Start.ToString (), ap.Duration.ToString (),
          ap.Sensitivity.ToString ()));
      }
    }
    return alListe;
  } else
    throw new MobileException ("oS does not exist");
}
public Business.AppointmentVO getCurrentAppointment ()
{
  //Shared resources
  Business.AppointmentVO appVo = null;
//New implementation
  Appointment currentApp = Microsoft.WindowsMobile.Status.
    SystemState.CalendarAppointment;// CalendarNextAppointment;
  appVo = new Outlook_1.Business.AppointmentVO (currentApp);
  if (appVo == null)
    return new Business.AppointmentVO ();//return new empty
  else
    return appVo;
}
```

Excerpt from ZoneVO.cs

```
public double ComputeLatLonDistance (double Lat1,
              double Long1, double Lat2, double Long2)
{
  /*The Haversine formula according to:
  http://mathforum.org/library/drmath/view/51879.html
        */
  double dDistance = Double.MinValue;
  double dLat1InRad = Lat1 * (Math.PI / 180.0);
  double dLong1InRad = Long1 * (Math.PI / 180.0);
  double dLat2InRad = Lat2 * (Math.PI / 180.0);
  double dLong2InRad = Long2 * (Math.PI / 180.0);
  double dLongitude = dLong2InRad - dLong1InRad;
  double dLatitude = dLat2InRad - dLat1InRad;
  // Intermediate result a.
  double a = Math.Pow (Math.Sin (dLatitude / 2.0), 2.0) +
                  Math.Cos (dLat1InRad) * Math.Cos (dLat2InRad) *
  Math.Pow (Math.Sin (dLongitude / 2.0), 2.0);
  // Intermediate result c (great circle distance in Radians).
  double c = 2.0 * Math.Atan2 (Math.Sqrt (a), Math.Sqrt (1.0 -a));
  // Distance
  const Double kEarthRadiusKms = 6371;
  dDistance = kEarthRadiusKms * c;
  return dDistance;
}
```

Excerpt from Startup.cs

```
private void Startup_Load (object sender, EventArgs e)
{
  updateDataHandler = new EventHandler (UpdateData); //This calls the UpdateData(...)
  this.Width = Screen.PrimaryScreen.WorkingArea.Width;
  this.Height = Screen.PrimaryScreen.WorkingArea.Height;
  gps.DeviceStateChanged += new
  GpsLib.DeviceStateChangedEventHandler (gps_DeviceStateChanged);
  gps.LocationChanged += new
  GpsLib.LocationChangedEventHandler (gps_LocationChanged);
  //sms
  sms = new MessageInterceptor (InterceptionAction.NotifyAndDelete);
  sms.MessageCondition = new MessageCondition ();
  sms.MessageReceived += new MessageInterceptorEventHandler (sms_MessageReceived);
}
```

```
public Startup ()
{
  InitializeComponent ();
  InitListViewComponent ();
  InitZoneComponent ();
  lblSetGps.Text = gps.GetDeviceState().
    DeviceState.ToString ();
  //Following two lines handles entry of new Appointment
  calendarNewAppointment = new SystemState
    (SystemProperty.CalendarAppointment);
  calendarNewAppointment.Changed += new
  ChangeEventHandler (calendarApp_Changed);
}
```

Appendix H

Source Code Excerpt Experiment 2

For detailed source code listing, please visit:

https://github.com/tormorten/brunelPhD/tree/master/experiment_2

Excerpt from Android participant client:

Excerpt from PresentationNotesReaderImpl.java

```
@Override
//Reading Meting Notes
public void run() {
    String prevNote = "";
    HttpPost post = new HttpPost(serverAddress);

    while (continueReading.get()) {
        try {
            Note outputNote = Note.newBuilder().setBtAddress(bluetoothAddress).build();
            post.setEntity(new ByteArrayEntity(outputNote.toByteArray()));

            HttpResponse response = httpClient.execute(post);
            HttpEntity responseEntity = response.getEntity();
            InputStream input = responseEntity.getContent();

            Note note = Note.parseFrom(input);

            if (!prevNote.equals(note.getMessage())) {
                Log.d(TAG, "Note received: " + note.toString());

                Bundle noteBundle = new Bundle();
                noteBundle.putString(BUNDLE_KEY, note.getMessage());
                Message message = handler.obtainMessage();
                message.setData(noteBundle);

                handler.sendMessage(message);
            }

            prevNote = note.getMessage();
        } catch (IOException io) {
            Log.e(TAG, "IOException when receiving note from " + serverAddress, io);

            Bundle noteBundle = new Bundle();
            noteBundle.putString(BUNDLE_KEY, "IOException: " + io.getMessage());
            Message message = handler.obtainMessage();
            handler.sendMessage(message);
            continueReading.set(false);
        }

        try {
            Thread.sleep(1000);
        } catch (InterruptedException ie) {
        }
    }
```

Excerpt from PresentationServiceImpl.java

```
public String getNextNote() {
    String note = null;

    if (historySlideNumber < (meetingNotes.size() - 1)) {
        historySlideNumber++;
    }

    if (historySlideNumber <= meetingNotes.size()) {
        note = meetingNotes.get(historySlideNumber);
    }

    return note;
}
```

...

```
 public void onCreate(Bundle savedInstanceState) {
     super.onCreate(savedInstanceState);
     Injector injector = Guice.createInjector(new MeetingRoomModule());
     presentationService = injector.getInstance(PresentationService.class);

     setContentView(R.layout.main);
 }
```

From ParticipantClientMain.java

```
private String getSavedServerAddress() {
    if (defaultServerAddress == null || "".equals(defaultServerAddress)) {
        defaultServerAddress = getString(R.string.defaultServerAddress);
    }

    SharedPreferences sharedPreferences = getSharedPreferences(PreferencesConstants.
            PREF_NETWORK_CONFIG, Activity.MODE_PRIVATE);
    String serverAddress = sharedPreferences.getString(PreferencesConstants.
            KEY_SERVER_ADDRESS, defaultServerAddress);

    Log.d(TAG, "Server address: " + serverAddress);

    return serverAddress;
}
```

Excerpt from Windows Mobile participant client

Excerpt from Presentation.cs

```
private void mnuNext_Click(object sender, EventArgs e)
{
    normalPresentation = false;

    if (pointer == meetingNotes.Count)
        presentNote((Note)meetingNotes[meetingNotes.Count-1]);
    else
    {
        presentNote((Note)meetingNotes[pointer]);
        pointer--;
    }
}

private void mnuPrev_Click(object sender, EventArgs e)
{
    normalPresentation = false;

    if (pointer <= 0)
        presentNote((Note)meetingNotes[0]);
    else
    {
        presentNote((Note)meetingNotes[pointer]);
        pointer--;
    }
}
```

Excerpt from Form1.cs

```
public byte[] Serialize(ac.uk.brunel.contextaware.Note obj)
{
    byte[] raw;
    using (MemoryStream memoryStream = new MemoryStream())
    {
        Serializer.Serialize(memoryStream, obj);
        raw = memoryStream.ToArray();
    }

    return raw;
}

public ac.uk.brunel.contextaware.Note Deserialize(byte[] serializedType)
{
    ac.uk.brunel.contextaware.Note obj;
    using (MemoryStream memoryStream = new MemoryStream(serializedType))
    {
        obj = Serializer.Deserialize<ac.uk.brunel.contextaware.Note>(memoryStream);
    }
    return obj;
}
```

```
private void connectToEngineStackoverFlow(string method, string directory, string data)
{

ac.uk.brunel.contextaware.Note myNote = new ac.uk.brunel.contextaware.Note();
myNote.btAddress = "0017833F9DF3"; //win mob sin

HttpWebRequest request = (HttpWebRequest)WebRequest.
  Create("http://context-aware-meeting-room.appspot.com/presentationNotes");
request.KeepAlive = false;
request.ProtocolVersion = HttpVersion.Version10;
request.Method = "POST";
request.UserAgent = "Windows Mobile";
request.AllowWriteStreamBuffering = true;

System.IO.Stream requestStream = request.GetRequestStream();

byte[] postBytes = this.Serialize(myNote);
// now send it
requestStream.Write(postBytes, 0, postBytes.Length);
    requestStream.Flush();
    requestStream.Close();

HttpWebResponse response;

response = (HttpWebResponse)request.GetResponse();

System.IO.Stream responseStream = response.GetResponseStream();

 ac.uk.brunel.contextaware.Note myNoteRec = ProtoBuf.Serializer.
    Deserialize<ac.uk.brunel.contextaware.Note>(responseStream);
}
```

Excerpt from Android presenter client

Excerpt from PresenterServiceImpl.java

```
public class PresenterServiceImpl implements PresenterService {
    private final ConnectionPing connectionPing;
    private final PresenterEventSender presenterEventSender;

    @Inject
    public PresenterServiceImpl(final ConnectionPing connectionPing, final PresenterEventSender presenterEventSender) {
        this.connectionPing = connectionPing;
        this.presenterEventSender = presenterEventSender;
    }

    /* (non-Javadoc)
     * @see ac.uk.brunel.mobile.contextaware.business.PresenterService2#initService(java.lang.String, ac.uk.brunel.mobile.
     */
    public void initService(final String serverAddress, final PresentationCallback presentationCallback) {
        connectionPing.setServerAddress(serverAddress);
        connectionPing.setPresentationCallback(presentationCallback);

        presenterEventSender.setServerAddress(serverAddress);
        presenterEventSender.setPresentationCallback(presentationCallback);
    }

    /* (non-Javadoc)
     * @see ac.uk.brunel.mobile.contextaware.business.PresenterService2#sendConnectionPing()
     */
    public void sendConnectionPing() {
        connectionPing.sendConnectionPing();
    }

    /* (non-Javadoc)
     * @see ac.uk.brunel.mobile.contextaware.business.PresenterService2#sendNextSlideEvent()
     */
    public void sendNextSlideEvent() {
        presenterEventSender.sendNextSlideEvent();
    }

    /* (non-Javadoc)
     * @see ac.uk.brunel.mobile.contextaware.business.PresenterService2#sendPrevSlideEvent()
     */
    public void sendPrevSlideEvent() {
        presenterEventSender.sendPrevSlideEvent();
    }
}
```

Excerpt from PresenterService.java

```
public interface PresenterService {
    public void initService(final String serverAddress,
                            final PresentationCallback presentationCallback);
    public void sendConnectionPing();
    public void sendNextSlideEvent();
    public void sendPrevSlideEvent();
}
```

Excerpt from PresenterEventSenderThread.java

```
private static class PresenterEventSenderThread implements Runnable {
    private static final String TAG = PresenterEventSenderThread.class.getName();
    private final PresenterAction presenterAction;
    private final HttpClient httpClient;
    private final Handler handler;

    private PresenterEventSenderThread(final PresenterAction presenterAction,
                                       final HttpClient httpClient, final Handler handler) {
        this.presenterAction = presenterAction;
        this.httpClient = httpClient;
        this.handler = handler;
    }

    @Override
    public void run() {
        try {
            HttpPost post = new HttpPost(serverAddress);

            BasicHttpEntity entity = new BasicHttpEntity();
            entity.setContent(new ByteArrayInputStream(presenterAction.toByteArray()));
            post.setEntity(entity);

            httpClient.execute(post);
        } catch (IOException io) {
            String errorMsg = "Unable to send the PresenterAction object, " + io.getMessage();
            // adding stack trace causes problems with the log
            Log.e(TAG, errorMsg);

            Bundle bundle = new Bundle();
            bundle.putString(BUNDLE_KEY, errorMsg);
            Message message = handler.obtainMessage();
            message.setData(bundle);

            handler.sendMessage(message);
        }
    }
}
```

```
private void sendPresentationEvent(final String event) {
    Log.d(TAG, "Sending presentationEvent: " + event);

    threadHandler = new Handler() {
        @Override
        public void handleMessage(Message msg) {
            presentationCallback.setStatusMessage(msg.getData().
                    getString(BUNDLE_KEY));
        }
    };

    final PresenterAction presenterAction = PresenterAction.
            newBuilder().setAction(event).build();

    executor.execute(new PresenterEventSenderThread(presenterAction,
            httpClient, threadHandler));
}
```

Excerpt from JavaME participant client:

Excerpt from BluetoothDevicePing.java

```
private final int serverPollFrequency;
private final NetworkCommunicationUtil networkCommunicationUtil;
private MeetingInput controller;
// Package visibility for testing purposes
CommunicationThread communicationThread = null;

public MeetingNoteReaderImpl(final int serverPollFrequency, final NetworkCommunicationUtil
    networkCommunicationUtil) {
    this.serverPollFrequency = serverPollFrequency;
    this.networkCommunicationUtil = networkCommunicationUtil;
}

public void setMeetingInput(final MeetingInput controller) {
    this.controller = controller;
}

public void stopCommunication() {
    if (communicationThread != null) {
        if (logger.isDebugEnabled()) {
            logger.debug("Stopping network communication");
        }

        communicationThread.stopCommunication();
    }
}

public void startServerCommunication(final String bluetoothAddress, final String serverAddress)
    {
    if (logger.isDebugEnabled()) {
        logger.debug("Starting network communication, server address: " + serverAddress);
    }

    communicationThread = new CommunicationThread(bluetoothAddress, serverAddress, controller,
        networkCommunicationUtil, serverPollFrequency);
    new Thread(communicationThread).start();
}
```

```
private void parseFromStream(final InputStream input) throws IOException {
    if (input.available() > 0) {
        final Note note = Note.parseFrom(input);

        if (meetingId == null || "".equals(meetingId)) {
            meetingId = note.getMeetingId();
        }

        if (latestMessage == null || !latestMessage.equals(note.getMessage())) {
            latestMessage = note.getMessage();

            if(logger.isDebugEnabled()) {
                logger.debug("Note received: " + note.toString());
            }

            controller.setNote(note.getMeetingId(), note.getMessage());
        }
    }
}

private void updateTimeoutCounter(final String errorMsg) {
    serverTimeoutCounter++;

    if(serverTimeoutCounter == 7) {
        controller.setErrorMessage("Error parsing Note object, " + errorMsg);
        continueReading = false;
    }
}
```

```
public void run() {
    while (continueReading) {
        if (logger.isDebugEnabled()) {
            logger.debug("Polling server: " + serverAddress);
        }

        HttpConnection connection = null;
        OutputStream output = null;
        InputStream input = null;

        try {
            connection = networkCommunicationUtil.createConnector(serverAddress);
            output = connection.openOutputStream();
            writeToStream(output);

            input = connection.openInputStream();
            parseFromStream(input);

            if(logger.isDebugEnabled()) {
                logger.debug("Response message: " + connection.getResponseMessage());
            }
        } catch (IOException io) {
            if (logger.isErrorEnabled()) {
                logger.error("IOException when parsing the Note Protobuf message", io);
            }

            updateTimeoutCounter(io.getMessage());
        } finally {
            if (output != null) {
                try {
                    output.close();
                } catch (IOException io) {
                }
            }
            if (input != null) {
                try {
                    input.close();
                } catch (IOException io) {
                }
            }
            if (connection != null) {
                try {
                    connection.close();
                } catch (IOException io) {
                }
            }
        }

        sleepForServerPollFrequency();
    }
}

private void writeToStream(final OutputStream output) throws IOException {
    final Note.Builder noteBuilder = Note.newBuilder().setBtAddress(bluetoothAddress);

    if (meetingId != null && !"".equals(meetingId)) {
        noteBuilder.setMeetingId(meetingId);
    }

    final Note note = noteBuilder.build();
    if(logger.isDebugEnabled()) {
        logger.debug("Writing note to stream: " + note);
    }

    note.writeTo(output);
    output.flush();
}
```

Excerpt from iPhone participant client:

Excerpt from context_aware_meeting_room_client_iphoneViewController.m

```
-(void)backgroundWork{
    [NSThread sleepForTimeInterval:1];

    counter = 0;
    historyCounter = 0;
    notes = [[NSMutableArray alloc] initWithCapacity:10];

    NSString *lastNoteMessage = @"";

    NSLog(@"Starting background thread");

    while(true){
        Note *requestNote = [[[Note builder] setBtAddress:@"012345678900"]build];
        NSURL *url = [NSURL URLWithString:@"http://context-aware-meeting-room.appspot.com/
            presentationNotes"];
        ASIHTTPRequest *request = [ASIHTTPRequest requestWithURL:url];
        [request addRequestHeader:@"User-Agent" value:@"iPhone"];
        [request appendPostData:[requestNote data]];
        [request setRequestMethod:@"POST"];

        [request startSynchronous];
        NSError *error = [request error];

        if(error) {
            NSLog(@"Error in network communication %@", [error userInfo]);
        } else {
            NSData *data = [request responseData];
            Note *receivedNote = [Note parseFromData:data];
            NSString *tempMessage = [receivedNote message];

            if(![lastNoteMessage isEqualToString:tempMessage]) {
                lastNoteMessage = tempMessage;
                [notes addObject:lastNoteMessage];
                counter++;
                historyCounter = (counter - 1);

                NSLog(@"Received note: %@", lastNoteMessage);

                [self performSelectorOnMainThread:@selector(updateMessageNoteText:) withObject:
                    lastNoteMessage waitUntilDone:YES];
            } else {
                NSLog(@"Same note message received");
            }
        }

        [NSThread sleepForTimeInterval:1];
    }

    [notes release];
}

- (void)updateMessageNoteText:(NSString *)lastNoteMessage {
    self.noteMessage.text = lastNoteMessage;
}
```

Excerpt from context_aware_meeting_room_client_iphoneViewController.m continued:

```
- (IBAction)nextButtonPushed:(id)sender {
    NSLog(@"Next button pushed");

    if(counter > 0) {
        if(historyCounter < (counter - 1)) {
            historyCounter++;
        }

        NSString *tempNote = [notes objectAtIndex:historyCounter];
        NSLog(@"next note: %@", tempNote);
        [self updateMessageNoteText:tempNote];
    }
}

- (IBAction)prevButtonPushed:(id)sender {
    NSLog(@"Prev button pushed");

    if(counter > 0) {
        if(historyCounter > 0) {
            historyCounter--;
        }

        NSString *tempNote = [notes objectAtIndex:historyCounter];
        NSLog(@"prev note: %@", tempNote);
        [self updateMessageNoteText:tempNote];
    }
}
```

Excerpt from local server:

Excerpt from BluetoothDevicePing.java

```
public interface BluetoothDevicePing {
    public void startSearch(List<String> registeredBluetoothDevices);
    public void stopSearch();
}
```

Excerpt from BluetoothDevicePingImpl.java

```
public void run() {
    if (logger.isInfoEnabled()) {
        logger.info("Starting Bluetooth device search");
    }

    int counter = 0;
    while (continueSearch.get() && bluetoothDevices.size() > 0) {
        try {
            if (counter >= bluetoothDevices.size()) {
                counter = 0;
            }

            devicePingCompleted.set(false);
            String btAddress = bluetoothDevices.get(counter);

            LocalDevice.getLocalDevice().getDiscoveryAgent().searchServices(null,
                    new UUID[]{BT_SERVICE}, new BluetoothDeviceImpl(btAddress), this);

            if (logger.isDebugEnabled()) {
                logger.debug("Searching for: " + btAddress);
            }


            while (!devicePingCompleted.get()) {
                synchronized (lock) {
                    if (!devicePingCompleted.get()) {
                        lock.wait();
                    }
                }
            }
        } catch (InterruptedException ie) {
            if (logger.isWarnEnabled()) {
                logger.warn("Interrupted exception when waiting for device search thread to notify", ie);
            }
        } catch (BluetoothStateException bse) {
            if (logger.isErrorEnabled()) {
                logger.error("Bluetooth exception during service search", bse);
            }
        } finally {
            counter++;
        }
    }

    if (bluetoothDevices.size() == 0) {
        bluetoothDeviceHandler.bluetoothDevicePingFinished();

        if (logger.isInfoEnabled()) {
            logger.info("No more registered participants, ending search for BluetoothDevices");
        }
    }
```

```
private synchronized void removeFoundBluetoothDeviceFromPingList(String foundBtAddress) {
    foundBtAddress = foundBtAddress.toUpperCase();

    for (int i = 0; i < bluetoothDevices.size(); i++) {
        String btAddress = bluetoothDevices.get(i).toUpperCase();

        if (btAddress.equals(foundBtAddress)) {
            bluetoothDevices.remove(i);

            if (logger.isDebugEnabled()) {
                logger.debug("Removed " + foundBtAddress + " from Bluetooth devices list");
            }

            break;
        }
    }
}
```

```
public void servicesDiscovered(int transactionId, ServiceRecord[] serviceRecords) {
    if (serviceRecords.length > 0) {
        final RemoteDevice remoteDevice = serviceRecords[0].getHostDevice();

        if (logger.isDebugEnabled()) {
            logger.debug("Bluetooth device and service found: " + remoteDevice.getBluetoothAddress());
        }

        final BluetoothDevice bluetoothDevice = createBluetoothDevice(remoteDevice);
        bluetoothDeviceHandler.addFoundDevice(bluetoothDevice);
        removeFoundBluetoothDeviceFromPingList(bluetoothDevice.getBluetoothAddress());
    }
}
```

Excerpt from cloud server:

Excerpt from PresentationIntegrationImpl.java

```
public MeetingNote createPresentationNoteObject(final Meeting meeting) {
    this.meeting = meeting;
    this.meetingNote = new MeetingNote.Builder(meeting.getMeetingId());

    BufferedReader reader = null;
    try {
        final String presentationLink = meeting.getPresentationLink();

        if (presentationLink != null && !"".equals(presentationLink)) {
            final URL url = new URL(presentationLink);
            reader = new BufferedReader(new InputStreamReader(url.openStream()));

            StringBuilder inputPage = new StringBuilder();
            String inputLine;

            while ((inputLine = reader.readLine()) != null) {
                inputPage.append(inputLine);
            }

            if (inputPage.length() > 0) {
                parseAndAppendPresentationNotes(inputPage.toString());
            }
        }
    } catch (MalformedURLException mal) {
        if (logger.isErrorEnabled()) {
            logger.error("Malformed presentationLink url: " + meeting.
                getPresentationLink());
        }
    } catch (IOException io) {
        if (logger.isErrorEnabled()) {
            logger.error("IOException when reading input line from " + meeting.
                getPresentationLink());
        }
    } finally {
        if (reader != null) {
            try {
                reader.close();
            } catch (IOException io) {
            }
        }
    }

    return meetingNote.build();
}
```

```
void parseAndAppendPresentationNotes(final String inputLine) {
    if (inputLine.contains(PresentationTags.START_TAG) && inputLine.contains
        (PresentationTags.END_TAG)) {
        String[] messageArray = inputLine.split(PresentationTags.NOTE_REGEX);
        String[] idArray = inputLine.split(PresentationTags.ID_REGEX);

        if (messageArray.length == idArray.length && messageArray.length >= 2) {
            for (int i = 0; i < messageArray.length; i++) {
                final String message = messageArray[i];
                final String id = idArray[i];
                final int endIndex = id.indexOf("&gt;");

                // Make sure the id is included in the start tag
                if (endIndex > -1) {
                    String number = id.substring(0, endIndex).trim();
                    // Find the number and continue if the end tag is included, also
                        makes sure id is a number
                    if (message.contains(PresentationTags.END_TAG) && number.matches
                        ("[0-9]*")) {
                        notes.add(message.substring(0, message.indexOf
                            (PresentationTags.END_TAG)));
                        slideNumbers.add(Integer.parseInt(number));
                    }
                }
            }

            // Finally add the notes to the meeting
            appendPresentationNotes();
        }
    }
}
```

Excerpt from CalendarIntegrationImpl.java

```
private List<Meeting> populateMeetingList(final List<Meeting> meetings, final List<
    CalendarEventEntry> calendarEventEntries) {
    for (CalendarEventEntry entry : calendarEventEntries) {
        final String meetingId = entry.getId();
        final String presenter = entry.getAuthors().get(0).getEmail();
        final List<String> participants = new ArrayList<String>();

        final List<EventWho> eventWhoList = entry.getParticipants();
        for (EventWho participant : eventWhoList) {
            // Do not add the presenter as a participant
            if (!presenter.equals(participant.getEmail())) {
                participants.add(participant.getEmail());
            }
        }

        final String title = entry.getTitle().getPlainText();
        final DescriptionInfo descriptionInfo = new DescriptionInfo(entry.
            getTextContent().getContent().getPlainText());
        final String description = descriptionInfo.getDescription();
        final String presentationLink = descriptionInfo.getPresentationLink();

        final List<Where> whereList = entry.getLocations();
        final StringBuilder locationBuilder = new StringBuilder();

        for(int i = 0; i < whereList.size(); i++) {
            locationBuilder.append(whereList.get(i).getValueString());

            if(i < whereList.size() -1) {
                locationBuilder.append(", ");
            }
        }

        final String startTime = entry.getTimes().get(0).getStartTime().toUiString();
        final String endTime = entry.getTimes().get(0).getEndTime().toUiString();

        meetings.add(new Meeting(meetingId, presentationLink, presenter, participants,
            title, description, locationBuilder.toString(), startTime, endTime));
    }

    return meetings;
}
```

```
public List<Meeting> getMeetingList(final Date fromDate, final Date toDate) {
    final List<Meeting> meetings = new ArrayList<Meeting>();
    calendarLogon.accountLogon(calendarService);

    String feedUrl = PropertiesReader.SERVER_APP.get(PropertiesConstants.
        CALENDAR_FEED_URL);

    try {
        CalendarQuery query = new CalendarQuery(new URL(feedUrl));

        query.setMinimumStartTime(new DateTime(fromDate));
        query.setMaximumStartTime(new DateTime(toDate));

        CalendarEventFeed feed = calendarService.query(query, CalendarEventFeed.class);
        if (logger.isInfoEnabled()) {
            logger.info("Number of meetings found: " + feed.getEntries().size());
        }

        populateMeetingList(meetings, feed.getEntries());
    } catch (IOException io) {
        if (logger.isErrorEnabled()) {
            logger.error("IOException when running query on feedUrl: " + feedUrl, io);
        }
    } catch (ServiceException se) {
        if (logger.isErrorEnabled()) {
            logger.error("ServiceException when running query on feedUrl: " + feedUrl,
                se);
        }
    }

    return meetings;
}
```

Appendix I

Source Code Excerpt Experiment 3

For detailed source code listing, please visit:

https://github.com/tormorten/brunelPhD/tree/master/experiment_3

Excerpt from Android client application

Excerpt from AppDashboardController.java

```
@Override
public void onCreate(Bundle savedInstanceState) {
    setContentView(R.layout.appdashboard);
    super.onCreate(savedInstanceState);

}

@Override
protected void initComponents() {
    lstApps = (ListView) findViewById(R.id.lstApps);
    gridApps = (GridView) findViewById(R.id.gridApps);
}

@Override
protected void renderView() {
    applications = new ArrayList<ApplicationDto>(Arrays.asList(
            new ApplicationDto("Calendar", R.drawable.calendar_app_b),
            new ApplicationDto("Gmail", R.drawable.gmail_app),
            new ApplicationDto("Settings", R.drawable.settings_app),
            new ApplicationDto("Gallery", R.drawable.gmail_app),
            new ApplicationDto("Comming", R.drawable.not_avail),
            new ApplicationDto("Comming", R.drawable.not_avail),
            new ApplicationDto("Comming", R.drawable.not_avail)));

    lstApps.setAdapter(new ApplicationsAdapter(context, applications));

    gridApps.setAdapter(new ApplicationsAdapter(context, applications));
}
```

Excerpt from SensorMonitor.java

```
@Override
public void onSensorChanged(SensorEvent sensorEvent) {
    StringBuilder sb = new StringBuilder();

    for (int i = 0; i < sensorEvent.values.length; i++) {
        float sensorValue = sensorEvent.values[i];

        if (sensorItem.getNumberOfLabels() > i) {
            sb.append(sensorItem.getLabelText(i)).append("\n");
            sb.append(decimalFormat.format(sensorValue)).append("\n\n");
        }

        if (sensorItem.getSensor().getType() == Sensor.TYPE_LIGHT) {
            changeBackground(sensorValue);
        }
    }

    TextView statusText = (TextView) findViewById(R.id.statusText);
    statusText.setText(sb.toString());
}
```

Excerpt from SensorListActivity.java

```
public void initAllSensors() {
    SensorManager sensorManager = (SensorManager) getSystemService(SENSOR_SERVICE);
    List<Sensor> sensors = sensorManager.getSensorList(Sensor.TYPE_ALL);

    Log.i(TAG, "Sensors found: " + sensors.toString());

    for (Sensor sensor : sensors) {
        SensorItem.setSensor(sensor);
    }

    arrayAdapter = new ArrayAdapter<SensorItem>(this,
            R.layout.sensor_row,
            R.id.row_text,
            SensorItem.values());

    setListAdapter(arrayAdapter);
}
```

Excerpt from Server application:

Excerpt from Cloud2DevicePushImpl.java

```
private final AppEngineAuthentication appEngineAuthentication;
private final DeviceIdRetrieverDao deviceIdRetrieverDao;
private final URL c2dmSendUrl;

@Inject
public Cloud2DevicePushImpl(final AppEngineAuthentication appEngineAuthentication,
                            final DeviceIdRetrieverDao deviceIdRetrieverDao) {
    this.appEngineAuthentication = appEngineAuthentication;
    this.deviceIdRetrieverDao = deviceIdRetrieverDao;

    try {
        c2dmSendUrl = new URL(PropertiesLoader.APP.getProperty(HomeScreenConstants.
            PROP_C2DM_SEND_URL));
    } catch (MalformedURLException mal) {
        logger.error("Invalid c2dm send url in application.properties", mal);
        throw new IllegalStateException("Invalid c2dm send url in application.
            properties");
    }
}

public void pushConfiguration(final UserConfiguration userConfiguration) {
    final String token = appEngineAuthentication.getAuthToken();

    DeviceId deviceId = deviceIdRetrieverDao.getDeviceId(userConfiguration.getUser())
        ;

    if (deviceId == null) {
        logger.error("DeviceId not found in datastore, push message will not be sent"
            );
    } else {
        logger.info("Sending configuration to: " + deviceId.getEmail());
        logger.info("App Engine Token: " + token);
        logger.info("RegistrationId: " + deviceId.getRegistrationId());

        if (deviceId != null && deviceId.getRegistrationId().length() > 0) {
            sendMessage(token, deviceId, buildMessage(userConfiguration));
        }
    }
}


private String buildMessage(final UserConfiguration userConfiguration) {
    StringBuilder message = new StringBuilder();
    message.append("data.stats=").append(System.currentTimeMillis()).append("&");
    message.append(userConfiguration.toMessageString());

    return message.toString();
}
```

```
private void sendMessage(final String token, final DeviceId deviceId, final String
    message) {
    try {
        HttpURLConnection conn = (HttpURLConnection) c2dmSendUrl.openConnection();

        StringBuilder content = new StringBuilder();
        content.append("registration_id=").append(deviceId.getRegistrationId());
        content.append("&collapse_key=").append(System.currentTimeMillis()).append(
            "&").append(message);

        conn.setDoOutput(true);
        conn.setUseCaches(false);
        conn.setRequestMethod("POST");
        conn.setRequestProperty("Content-Type", "application/x-www-form-urlencoded");
        conn.setRequestProperty("Content-Length", Integer.toString(content.length()))
            ;
        conn.setRequestProperty("Authorization", "GoogleLogin auth=" + token);
        conn.setRequestProperty("", deviceId.getRegistrationId());

        logger.info("sending message: " + content.toString());

        OutputStream out = conn.getOutputStream();
        out.write(content.toString().getBytes("UTF-8"));
        out.close();

        logger.info("" + conn.getResponseCode());
        logger.info(conn.getResponseMessage());

        // Check for updated token header
        String updatedAuthToken = conn.getHeaderField("Update-Client-Auth");
        if (updatedAuthToken != null && !token.equals(updatedAuthToken)) {
            logger.warn("Got updated auth token from C2DM servers: " +
                updatedAuthToken);
            logger.warn("Persisting updated token, please try to push message again")
                ;
            appEngineAuthentication.persistToken(updatedAuthToken);
        }
    } catch (Exception e) {
        logger.error("Error sending c2dm message", e);
    }
}
```