A Model for Context Awareness for Mobile Applications using Multiple-Input Sources

Direshin Pather

Supervisor: Prof. JL Wesson
Co-Supervisor: Dr NLO Cowley
Department of Computing Sciences

January 2015

Submitted in fulfilment of the requirements for the degree of
Magister Commercii (Computer Science and Information Systems) in the Faculty of Science at the Nelson Mandela Metropolitan University

Declaration

I, Direshin Pather, hereby declare that the dissertation for the degree Magister Commercii is my own work and that it has not previously been submitted for assessment or completion of any postgraduate qualification to another University or for another qualification.

Direshin Pather

Acknowledgements

I would like to take this opportunity to thank my supervisors, Prof. Janet Wesson and Dr Lester Cowley for their inspiring guidance, continuous encouragement, friendly advice and lively enthusiasm throughout this two year journey of continuous learning.

I would especially like to express my sincere gratitude to them for assisting and providing valuable feedback to improve this dissertation.

I would also like to thank the staff of the Department of Computing Sciences for their expertise and support throughout my research.

I would also like to thank the generous funders of this research, namely the Telkom / NMMU Centre of Excellence and THRIP.

Lastly, I would like to thank my family and close friends for their continuous and unconditional support and understanding throughout this research.

Thank you,
Direshin

Summary

Context-aware computing enables mobile applications to discover and benefit from valuable context information, such as user location, time of day and current activity. However, determining the users' context throughout their daily activities is one of the main challenges of context-aware computing. With the increasing number of built-in mobile sensors and other input sources, existing context models do not effectively handle context information related to personal user context.

The objective of this research was to develop an improved context-aware model to support the context awareness needs of mobile applications. An existing context-aware model was selected as the most complete model to use as a basis for the proposed model to support context awareness in mobile applications.

The existing context-aware model was modified to address the shortcomings of existing models in dealing with context information related to personal user context. The proposed model supports four different context dimensions, namely Physical, User Activity, Health and User Preferences. A prototype, called CoPro was developed, based on the proposed model, to demonstrate the effectiveness of the model. Several experiments were designed and conducted to determine if CoPro was effective, reliable and capable. CoPro was considered effective as it produced low-level context as well as inferred context. The reliability of the model was confirmed by evaluating CoPro using Quality of Context (QoC) metrics such as Accuracy, Freshness, Certainty and Completeness. CoPro was also found to be capable of dealing with the limitations of the mobile computing platform such as limited processing power.

The research determined that the proposed context-aware model can be used to successfully support context awareness in mobile applications. Design recommendations were proposed and future work will involve converting the CoPro prototype into middleware in the form of an API to provide easier access to context awareness support in mobile applications.

Key words: Context awareness, mobile applications, sensors, personal user context, Design Science Research.

Table of Contents

List of Figures

List of Tables

List of Equations

Chapter 1: Introduction

1.1 Background

Real-time access to context information can support time-critical applications, such as emergency healthcare and location-based services (Fisher & Monahan 2012). This is due to the extensive adoption of phones with powerful context awareness features enabled through sensors and personal information. Context awareness can be defined as the work that results in the automation of a software system, which is based on the information about the user's context (Dey & Abowd 1999). Another factor contributing to this trend is the tendency of individuals to carry their phones with them everywhere (Klasnja & Pratt 2011).

Context awareness is a key concept to achieve ubiquitous computing, which enables information technology to be invisible whilst still being integrated within our daily lives (Zhu *et al*. 1992). By facilitating context awareness in mobile devices such as smart phones, richer human-computer interaction and less usability issues can be achieved (Mihalic, Tscheligi & Unit 2006).

Dey and Abowd (2001) define context as any information that can be used to characterize the situation of an entity. An important aspect of context awareness that is needed to characterize the situation of an entity is location awareness.

Location awareness enables services to provide or access information relevant to the current situation such as a patient's location. Patient location is important contextual information that is required in healthcare systems and especially in remote health monitoring (Liu, Zhu, Holroyd & Seng 2011; Elgazzar, Aboelfotoh, Martin & Hassanein 2012). Location becomes a crucial attribute for patients who suffer from memory loss diseases such as Alzheimer’s disease. Having access to the patient's location can help to provide timely medical assistance in emergency and life-threatening situations (Bricon-Souf & Newman 2007).

Location-based services are an essential aspect of context awareness. A key factor in location awareness is the accuracy of location estimation (Lo *et al*. 2010). In the case of outdoor environments, the Global Positioning System (GPS) suffices as a location finding technique.

In the case of indoor environments, GPS is not sufficient as it does not give an accurate enough reading to be accurate, or even no reading. There are, however, several other new technologies and techniques being used such as using RFID and Wi-Fi signals. However, none of these stand out as a standard solution for locating indoor position accurately as they each have their own limitations (Kerr, Duncan, Schipperijn & Schipperjin 2011).

Mobile phones can make it possible to capture contextual information to help individuals to better understand the circumstances that affect their daily lives. Location, which is part of the environmental context of an entity, can usually be acquired by accessing the mobile phone's sensors such as the GPS (Google Inc 2014e). Other sources of contextual information can also include information in a user's calendar. Such information would help to develop the user's profile and could be used with other contextual information to help determine the user's context. However determining the user's context throughout their daily activities is one of the main challenges in this research area (Santos *et al*. 2009). Using multiple-input sources to identify and predict context for given situations, such as being at home could help to solve this problem as suggested by Mitchell (2011).

1.2 Problem Statement

Mobile applications do not currently incorporate multiple-input sources to accurately determine the context. Context is essential in cases such as anti-theft or near-emergency services (Santos *et al*. 2009). To provide these types of services mobile devices need to be able to clearly identify specific contexts of the user. Mobile devices with their increasing capabilities include sensors from which data such as position, lighting or sound can be obtained, which can help to determine the user's context.

Using sensors it could be possible to determine whether an elderly person has fallen at home and has been unable to move for a period of time, consequently triggering an emergency call. However, there are multiple sources of data with unique patterns that need to be captured and processed timeously, which is difficult (Santos *et al*. 2009). Other challenges include the battery life of a mobile device as the energy required by context sensors is significant and can drain the battery quickly (Rahmati, Shepard, Tossell, Zhong & Kortum 2012).

Barkhuus and Dey (2003) conducted a study on how participants evaluated three levels of interaction including personalization, passive context awareness and active context awareness. This study concluded that users are willing to accept a large degree of automation in applications as long as the application's usefulness is greater than the cost of limited control.

The main objective of this research is to develop a context-aware model to support the context awareness needs of mobile applications. This context-aware model will then be implemented as a prototype in order for the feasibility of the context-aware model to be evaluated. The prototype will be evaluated in terms of its effectiveness, reliability and capability. The prototype will be considered effective if it can produce different levels of context. The reliability of the model will be confirmed through evaluating the prototype using evaluation metrics (i.e. QoC metrics). The prototype's capabilities will also be assessed in terms of dealing with the limitations of the mobile computing platform such as limited battery power. However, in order to achieve this objective, developments and shortcomings related to context awareness in mobile applications need to be identified and understood. Existing solutions and literature related to context awareness will be reviewed in detail. This review will form the basis of facilitating context awareness in mobile applications and the development of the context-aware model.

1.3 Aim of Research

The aim of the research is to develop a context-aware model that can support context awareness in mobile applications using multiple-input sources.

1.4 Research Outline

This section provides an outline of how the research will be structured to meet the research objectives. The outline is described by identifying the research objectives, research questions, scope and constraints and the research methods, which will be used to address the research questions.

1.4.1 Research Questions

The following main research question will be addressed by this research:

> *How can an improved context-aware model be developed for mobile applications using multiple-input sources?*

The above research question will be answered by addressing the following sub-questions:

RQ 1. What are the context awareness problems and requirements of mobile applications?

RQ 2. What are the problems and requirements of existing context awareness solutions used in mobile applications?

RQ 3. How can an improved context-aware model be developed?

RQ 4. How effective, reliable and capable is the proposed context-aware model and to what extent does it support context awareness in mobile applications?

1.4.2 Research Objectives

The main research objective is thus:

> *To develop an improved context-aware model for mobile applications using multiple input sources.*

The sub-objectives that will assist in achieving the main research objective are:

RO 1. To identify the existing problems and requirements of context awareness in mobile applications (Chapter 2).

RO 2. To identify the existing problems of context awareness solutions that relate to mobile applications (Chapter 3)

RO 3. To develop a context-aware model using multiple input techniques and implement this model in a prototype (Chapter 4).

RO 4. To evaluate the utility, quality and efficacy of the prototype developed based on the proposed model (Chapter 5).

1.4.3 Research Methodology

This research will focus on the development and performance of the model and prototype. This relates to the main aim of developing a context-aware model for mobile applications, which will be applied to address the problems identified in context awareness. Thus, the Design Science Research (DSR) methodology will be used for the project (Hevner, March, Park & Ram 2004).

DSR is a methodology, which requires the creation of innovative, purposeful artefacts that address a specified problem in a specific problem domain. The artefacts constructed in DSR comprise of:

- Constructs - the language that helps to define and communicate the problems and solutions.
- Models - constructs used to represent a realistic scenario involving the problem design and corresponding solution space.
- Methods - solution processes that can vary from formal to informal.
- Instantiations - indications of how to implement constructs, models and methods in a functional system (Hevner, 2007).

These artefacts must be evaluated in order to ensure their utility (i.e. if it is a solution to the problem) for the specified problem. Furthermore the artefacts must also either solve a problem that has not yet been solved, or provide a more effective solution to produce an innovative research contribution.

Both the construction and evaluation of the artefacts must be done rigorously and the results of the research presented effectively. This is in order to demonstrate the extent to which the artefacts produced solve the identified problem in the specified application domain, which is the main goal of DSR.

Hevner *et al*. (2004) provide a set of seven guidelines (Table 1.1) which help conduct, evaluate and present DSR.

Table 1.1: A summary of the Design Science Research guidelines (Hevner *et al.* 2004)

Table 1.1. Design-Science Research Guidelines	
Guideline	**Description**
Guideline 1: Design as an Artefact	Design science research must produce a viable artefact in the form of a construct, a model, a method, or an instantiation.
Guideline 2: Problem Relevance	The objective of design-science research is to develop technology-based solutions to important and relevant business problems.
Guideline 3: Design Evaluation	The utility, quality, and efficacy of a design artefact must be rigorously demonstrated via well-executed evaluation methods.
Guideline 4: Research Contributions	Effective design-science research must provide clear and verifiable contributions in the areas of the design artefact, design foundations, and/or design methodologies.
Guideline 5: Research Rigor	Design-science research relies upon the application of rigorous methods in both the construction and evaluation of the design artefact.
Guideline 6: Design as a Search Process	The search for an effective artefact requires utilizing available means to reach desired ends while satisfying laws in the problem environment.
Guideline 7: Communication of Research	Design-science research must be presented effectively both to technology-oriented as well as management-oriented audiences.

These guidelines are based on procedures, principles and practices and address each of the key phases in the DSR process. These guidelines also facilitate good DSR practice and guarantee that rigorous and good quality deliverables are produced, for each phase of the DSR process.

Design Science Research is pragmatic in nature and is an embodiment of three closely related cycles of activities (Figure 1.1): the relevance, design and rigor cycles (Hevner 2007).

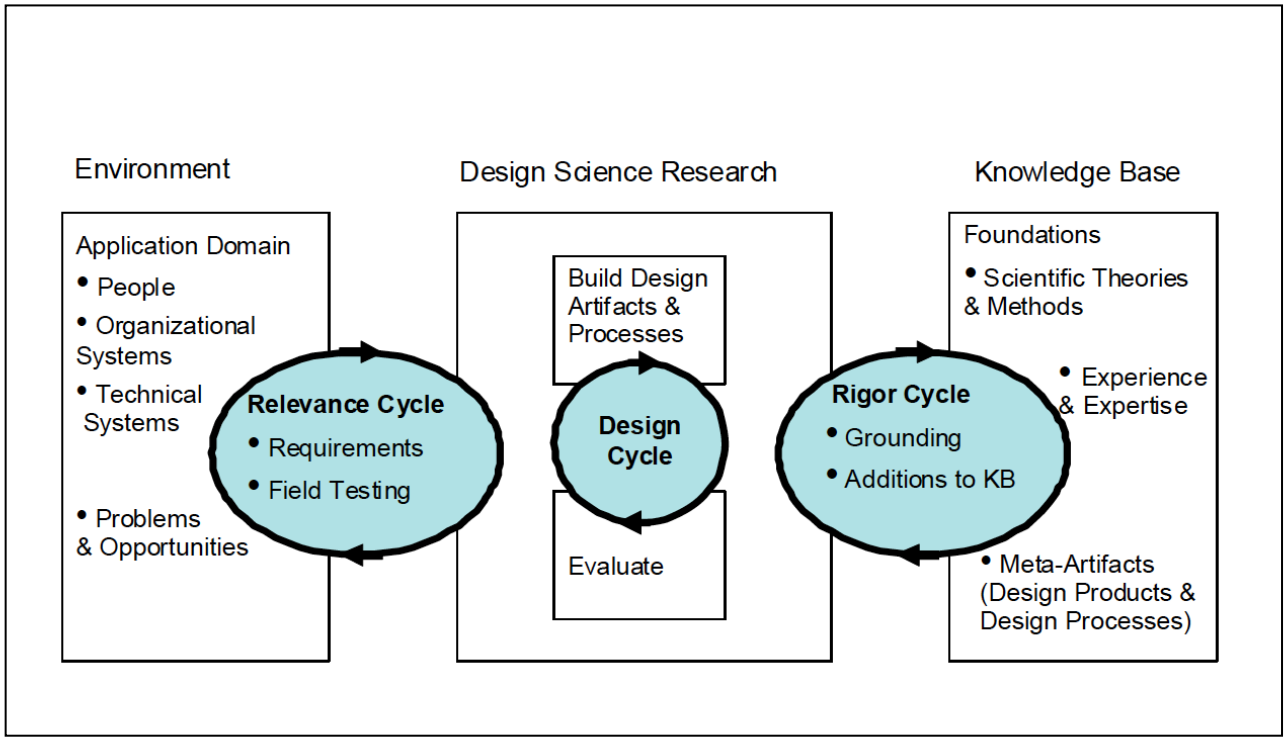

Figure 1.1 Design Science Research cycles (Hevner 2007)

Figure 1.1 above illustrates the three research cycles from the IS research framework, which include the relevance, rigor and design cycles (Hevner *et al*. 2004).

The Relevance Cycle initiates the DSR, identifying opportunities and problems that not only provide the input requirements for this research, but also identifying acceptance criteria for the final evaluation of the research results. The problem identified for the research is that mobile applications do not use multiple inputs to accurately determine context and therefore lack context awareness. The research results will be obtained when the research artefacts are introduced into the environmental field testing. This cycle will help to determine the requirements for the context-aware model and the criteria to be used during its evaluation (Hevner 2007).

Once the main problem has been identified, the Rigor Cycle can provide past knowledge to this research in the form of grounding theories and methods along with domain experience and expertise to ensure its innovation. The Rigor Cycle also enables new knowledge that is produced by this research to be added to the growing knowledge base (Hevner 2007). This cycle will help the literature studies to be done on the two knowledge bases of this research,

namely context awareness and mobile computing. It will also allow the results of the evaluation of the context-aware model to be added to the growing knowledge base.

The central Design Cycle supports the core activities of building and evaluating the context-aware model and prototype of this research. The Design Cycle iterates more rapidly between the related research activities and uses the generated feedback to refine the design of the model and prototype (Hevner 2007). This cycle will enable the design of the context-aware model and implementation of a mobile application based on the context-aware model.

Another crucial component of DSR, if not the most important aspect, is the proposed DSR contribution. The DSR contribution needs to meet several criteria, these include:

- Identification and clear description of the problems surrounding context awareness in mobile applications.
- Demonstration that no existing model in the present IT knowledge is sufficient.
- Construction and presentation of an innovative context-aware model (i.e. prototype) that addresses the lack of support for context awareness.
- The context-aware model (i.e. prototype) needs to be rigorously evaluated in order to assess its utility in the application domain.
- Communication of the value added from the results of evaluating the prototype to the IT knowledge-base and to practice.
- Description of the implications of the context-aware model (i.e. prototype). (Hevner *et al*. 2004)

1.4.4 Scope and Constraints

The focus of the research will be on developing a context-aware model by using multiple-input sources. The input sources and related input gathering techniques will be selected and used to develop a context-aware model. The input gathering techniques will be evaluated to establish which techniques will be the most appropriate techniques to adopt. An innovative artefact will then be developed using this context-aware model. The artefact will be evaluated in order to determine the effectiveness of the proposed context-aware model. The research will include developing a context-aware model and using this model to develop a mobile application to help determine the context of a user.

1.4.5 Envisaged Contribution

Context awareness in current mobile applications has been found to be limited and does not fully take multiple-input sources for context determination into account (David, Endler, Barbosa & Filho 2011). The main contribution of this research will be an improved artefact that will be designed and implemented to facilitate context awareness in mobile applications by using multiple-input sources. This artefact will be evaluated to determine whether it can facilitate context awareness in mobile applications and will provide a basis for future research in this area.

1.5 Dissertation Outline

The dissertation outline presents an overview of the contents of each chapter in a narrative descriptive form. The DSR methodology used will also assist in structuring the dissertation.

The introduction of the topic, problem identification and motivation for the research is covered in Chapter 1. This chapter also introduces several concepts specific to the research. The problem statement, aim of the research and research questions and objectives are addressed. The limitations of the research are addressed in order to determine and identify the scope and constraints. The use of DSR is highlighted. Envisaged contributions conclude the chapter.

Chapter 2 discusses the literature for the first DSR knowledge base (i.e. context awareness) by focusing on the mobile computing aspects of context awareness, including location awareness. The concept of context awareness is defined. Global Positioning Systems (GPS) and Indoor Positioning Systems (IPS) are discussed as part of location awareness. The existing issues and possible solutions of context awareness are also discussed to conclude the chapter.

The literature study in Chapter 3 covers the second DSR knowledge base (i.e. mobile computing), which discusses the application domain of mobile computing. Chapter 3 introduces and discusses the need for context awareness support within mobile applications. Context awareness in m-health as a possible sub-domain is discussed. Advantages and shortcomings of existing mobile context models are identified, which help justify the relevance of the research.

Chapter 4 addresses the design of the context-aware model. The context-aware model uses multiple input sources and was used to create a mobile application prototype to provide context information. The chapter discusses each of the components of the proposed model and how it was incorporated into an innovative prototype. The design processes involved in implementing the innovative prototype are discussed to conclude the chapter.

The evaluation of the model is discussed in Chapter 5. The evaluation methods and metrics used are selected to evaluate the objectives of the context-aware model. The context-aware model was evaluated by evaluating the context-aware application using the evaluation metrics. The experimental design used for the evaluation is discussed in detail. The results and analysis of these results are presented in this chapter. Design implications from the literature chapters are compared with the evaluation results. Design recommendations are made to conclude the chapter.

Chapter 6 discusses and presents the conclusions and contributions of the research. Conclusions are made as to whether the model can facilitate context awareness in mobile applications by using multiple inputs. Contributions of this research in terms of theoretical and practical contributions are recognized. The initial goals of the research are reviewed to determine whether the research met its projected objectives. An outline of the problems and limitations in conducting the research are also discussed. Lastly, future research is identified to conclude the chapter.

Chapter 2: Context Awareness

2.1 Introduction

This chapter together with the Relevance and Rigor Cycle, aims to answer Research Question 1: "*What are the context awareness problems and requirements of mobile applications?"* This chapter discusses the first DSR knowledge base by focusing on the mobile computing aspects of context awareness. An explanation of context and context awareness will be given. Determining the context of a mobile application and how context awareness is related to location awareness and ambient intelligence will be discussed. Each of the crucial design steps to facilitate context awareness in mobile applications will be dealt with in detail. Existing issues and possible solutions for context awareness will also be discussed to conclude the chapter.

2.1.1 Definition

Context awareness can be defined as:
"...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 & Abowd 1999).

This definition makes no assumptions about the types of information, which are relevant to context such as time, location, identity and activity.

Context is usually the location, identity and state of people, groups and computational and physical objects (Dey & Abowd 1999). Three entities including places, people and things were recognized by Dey, Abowd and Salbeer (2001). The term "places" refers to physical locations such as rooms, offices and buildings. People refers to individuals or groups. Things refer to tangible objects or software components (Debes, Lewandowska & Seitz 2005). To describe these entities, four categories were introduced by Debes *et al*. (2005) as follows :

- Identity – describes the entity with a clear identifier, which has to be unique in the domain of the application.
- Location – contains positioning data and orientation as well as information about regional relations to other entities such as neighboring entities. This comprises physical location data, i.e. geographical data, as well as spatial relations.

- Status – includes properties which can be perceived by a user. For a place, this can be, for example, the current temperature, the ambient lighting or the noise level. For persons this refers to physical factors like vital signs, tiredness or the current occupation.
- Time – is both date and time.

Another important dimension concerning context is activity, as it relates to the user's current task (Kaenampornpan 2004). Activity would refer to what is happening in the situation, for example the user is walking.

Context can be useful in terms of mobile users, if used correctly for the right person, at the right place and at the right time. However in order to use context in computing systems, these systems need to be context-aware.

Context awareness in computing can be understood as the existence of computer systems and applications, which can collect and understand "information about the immediate situation such as the people, roles, activities, times, places, devices, and software that define the situation" (Traynor, Xie & Curran 2010) . Based on this perceived context, the systems and applications should perform appropriate and related actions. These actions can involve the presentation of customized or specially formatted information or the performance of some action to avoid a potentially dangerous situation or assistance in the case of an emergency (Traynor *et al*. 2010). As a result context needs to be managed and interpreted correctly, as it may be acquired from multiple and heterogeneous sources.

Context awareness usually involves several complex steps including (Bessi & Bruni 2009; Demeester 2010; Hardian 2011):

- Acquisition of contextual information
- Monitoring contextual information
- Filtering contextual information
- Storing of contextual information
- Representation of contextual information
- Interpreting contextual information.

In order for context awareness to be effectively facilitated and managed, an underlying architecture needs to be in place.

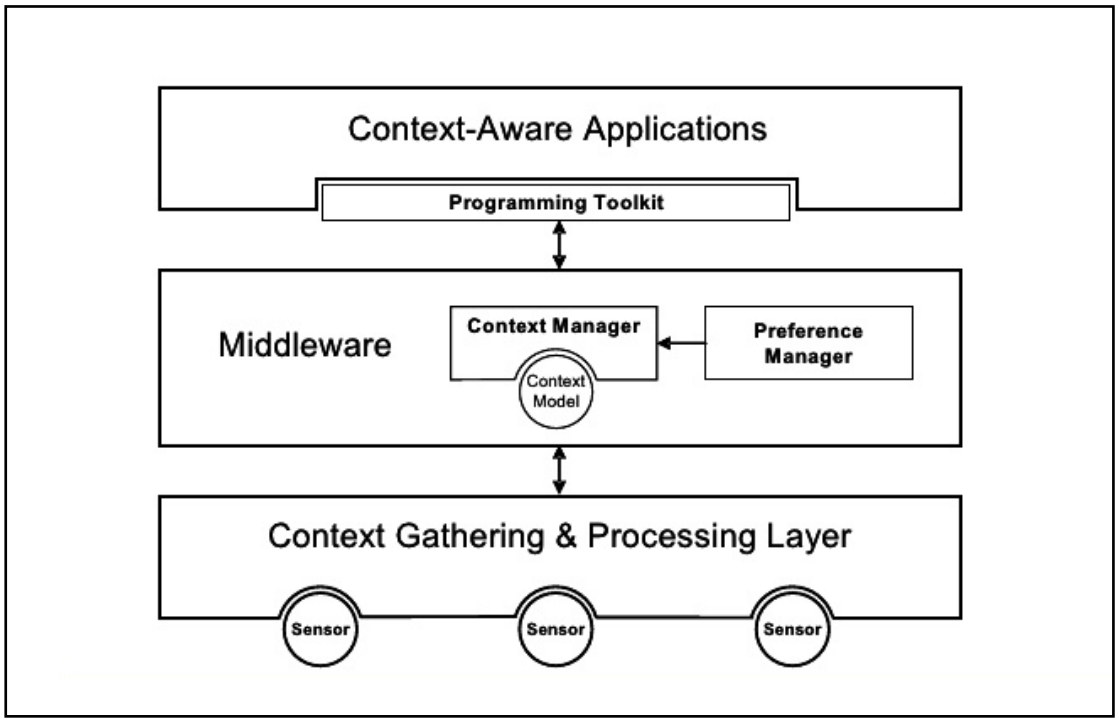

Figure 2.1: Typical architecture to support context-aware applications (Hardian 2011)

The typical architecture shown in Figure 2.1 includes elements such as the context gathering and processing layer, the middleware layer and the context-aware applications layer (Hardian 2011). The context gathering and processing layer, also known as the context provider, involves the acquiring, monitoring, filtering of contextual information. The middleware layer involves the storing, representation and interpreting of contextual information. The context manager stores and manages the context obtained from the context provider layer. The preference manager stores the user's preferences, which are used to tailor the context contained in the context manager. The context-aware application layer can be seen as the consumer layer, the layer in which context becomes an input for applications to use via the programming toolkit (i.e. application programming interface (API)), in order to appropriately change the application's behaviour.

Future work in the area of context awareness has the potential to significantly improve the way in which ubiquitous and intelligent computing environments support our everyday activities. Context awareness can also provide richer experiences in human-computer interaction and is closely related to location awareness and ambient intelligence (Traynor *et al*. 2010).

2.1.2 Location Awareness

An important sub-set of context awareness is location awareness. In order to provide context awareness, limitations in location awareness need to be addressed first in order to address context awareness.

Location awareness was primarily a notion of static user location (Trifa, Guinard & Mayer 2011); the concept was later extended to incorporate movement (Liu & Karimi 2006). Context models proposed by Bolchini *et al*. (2007) provide support for context-aware applications, which use location as a basis to adapt interfaces, refine application-relevant data, increase the accuracy of information retrieval, discover services, make user interaction implicit and build smart computing environments. For instance, a location-aware mobile device may verify that it is currently inside a building (Schmidt *et al*. 1999).

Users carry their mobile devices with them wherever they go, outdoors and indoors (Mitchell 2011). One of the unique functionalities available to mobile applications is location awareness. Thus, knowing the location of users and using this information appropriately can generate a more contextual experience to mobile users. Location awareness can essentially be understood as being the outdoor and indoor position and movement of a particular user or device. In order to obtain location awareness, two main technologies are used, namely Global Positioning Systems (GPS) and Indoor Positioning Systems (IPS).

GPS suffices as an outdoor location finding technique, but does not work very well when used in indoor environments such as buildings or heavily built-up environments. GPS requires consistent line-of-sight to orbiting satellites to avoid missing data (Kerr *et al*. 2011). This is a result of factors such as:

- Slow connectivity – If a user enters a building with a limited satellite view this can cause missing data or show that the user has not travelled at all.
- Physical structures – Satellite communication is interfered with by surrounding high buildings and building materials such as indoor locations.

GPS accuracy decreases (typical error of 40-50 m) when communication with satellites is compromised (Kerr *et al*. 2011). IPS tries to address this problem; however no standard

solution currently exists for determining indoor position accurately. Some indoor positioning solutions work similarly to GPS (Schutzberg 2013), such as:

- Locata offers beacons, which send signals that cover large areas and can penetrate walls. Locata receivers work similarly to GPS receivers.
- Nokia uses beacons that send Bluetooth signals. Any Bluetooth device can read these signals, but the signals from beacons only cover a few square meters.
- TruePosition offers a cell tower locating solution.
- Other solutions involve RFID tags and Wi-Fi signals.

No single solution works perfectly in all environments; the best solution for indoor and outdoor positioning may be a hybrid (Schutzberg 2013). Ambient intelligence can help to assist with this problem as it is characterized by systems and technologies that are embedded, personalized, context-aware, adaptive and anticipatory.

2.1.3 Ambient Intelligence

Ambient Intelligence (AmI) deals with the problem of how to create context-aware, computing environments which promote seamless human-computer interaction (Traynor *et al*. 2010). AmI incorporates the areas of ubiquitous computing, artificially intelligent systems and context awareness, among others.

Context awareness is an important function of AmI as context awareness facilitates AmI, by using sensors to communicate and help identify movements and actions (Traynor *et al*. 2010). Current mobile devices have the necessary sensors to support AmI and are frequently required to respond to changes in the environment, for example a change in location or context (Traynor *et al*. 2010).

Context and context awareness have been the main problems in AmI research in the past decade (Oh, Schmidt, Woo & Korea 2007; Lee, Lunney, Curran & Santos 2009). These problems involve issues such as combining inputs from multiple sensors in areas of reasoning and context. As a result, context awareness has been seen as a crucial concept in addressing automatic behaviours in pervasive and predictive systems. A key requirement for successful transparent interaction is up-to-date and valid context information. For example, a system that

senses a user's condition, location or physical actions and adapts to maximize user convenience is utilizing context awareness (Curran 2011).

There is currently a need for context-aware AmI systems to meet evolving user expectations by constantly and implicitly adapting to the surrounding environment (Curran 2011). This was highlighted by Lee and Chen (2009) whose research focused on discovering the latest developments in ubiquitous multimedia computing. The context-aware trends that were revealed were m-health, which includes health information systems, ubiquitous computing in health care and smart homes for older persons. Other context-aware areas include context awareness applications, context-aware computing technology in intelligent decision-making and context-aware proactive services.

Curran (2011) believes that context awareness and ambient intelligence will become more significant in the future. AmI, however, requires the development of innovative solutions that make use of context-aware technologies. These technologies are seen as a vital part of these developments in order to perceive valuable insight from the context and react upon it autonomously (Curran 2011).

AmI is presently being improved to build intelligent systems that support human activities in key problem domains, such as healthcare, ambient assisted living, and disaster recovery (Curran 2011). Access to accurate information is important and the pervasive nature of AmI technology ensures that users will have constant access to up-to-date information regardless of their location (Kosta, Pitkänen, Niemelä & Kaasinen 2010). In order to ensure that users have constant access to up-to-date information regardless of their context, their context needs to be acquired.

2.2 Acquiring Context

Context acquisition is the most basic level of context awareness. There are a number of ways of acquiring the context of a mobile application (Schmidt *et al*. 1999). One of these is that the mobile device itself can determine its own context (also known as sensed context). Another is that the network that the mobile device is connected to can determine the device's context. Integrating these two approaches, such as sensory data from the mobile device and the

network enables a clearer understanding of the context to be determined (Brander & Wesson 2007).

The first approach can be achieved by using the on-board sensors of a mobile device, either individually or by using multiple sensors, also known as sensor fusion (Mitchell, 2011). Sensor fusion involves combining sensors into more useful and abstract high-level sensors. These higher level sensors will provide more useful information. Examples of single sensors that could be used include the accelerometer, compass and gyroscope. An example of a higher level sensor is the rotation vector, which is an integration of the accelerometer, gyroscope and magnetometer. Other contextual data that can collectively help define the user, his/her behaviour and environment can include the user's calendar and preference data.

The second approach could involve the network providing context (downloaded context) via web services such as the weather or news (Mitchell 2011). The network approach could also be used to monitor user interaction and to listen for system events from the device. Context can also be explicitly provided; for example a user's preferences can be acquired directly from the user via the application user interface (Huang, Liu & Li 2011). Another approach could be to compute context information at run-time (i.e. derived context) (Huang *et al*. 2011).

Examples of derived context identified by Mitchell (2011) include:

- Movement - walking, running, driving.
- Location - home, work, travel
- Local environment - weather, ambient noise, ambient light
- Time of day, day of week

These derived contexts relate to context items identified by Antila, Polet and Sarjanoja (2011) which include:

- Activity – physical activity of the user
- Applications – currently open applications
- Device – device information, such as the device type
- Location – using GPS, network and Wi-Fi scan data, current street address and cell ID

- Surroundings – aspects of physical surroundings using ambient light detector and weather.

Information can also be acquired from machine-learning techniques, information extraction and retrieval programs (Lukowicz *et al*. 2012). There is a need for better hybrid knowledge representation and reasoning systems that can use logic-orientated representations such as OWL (Chen, Finin & Joshi 2003; Syed & Finin 2011).

In order to effectively determine the context of the user, multiple sources of inputs need to be used, such as sensors, calendar and web services. Multiple sources of inputs are needed to give a clearer picture of the context that is to be determined (Brander & Wesson 2007). Using multiple input sources requires the context to be monitored to ensure relevance and quality.

2.2.1 Monitoring Context

To actively support context awareness the multiple input sources of the users' context needs to be continuously monitored. Context monitoring is a process that involves continuously detecting changes in the users' context. This process requires continuous collection of data from the multiple input sources, processing the data to obtain context and managing changes to the users' context. This notion of context monitoring is different from the usual context recognition, which only identifies the current context. After a change in context is discovered, recognizing the context again would be redundant if the context has not changed. Context monitoring frequently involves several complex procedures such as feature extraction and context recognition spread across the multiple input sources all at the same time. Therefore, it is fairly difficult for each mobile application to perform the complicated context monitoring process on their own. The limited resources of mobile devices and the sharing of these resources amongst mobile applications support the idea of a central context-aware service to provide context information. To successfully support context monitoring as a part of facilitating context awareness, a central context monitoring solution is needed (Kang *et al*. 2010).

Context monitoring is used for measuring the relevance of the current context of an application. It can also support analysing the validity and quality of the context information it receives. Setting up mobile context monitoring involves dealing with complex, multi-

dimensional challenges that cover different technical and research issues. Applications need diverse contexts that differ in levels of awareness and accuracy. Users have unique requirements and preferences for context-aware applications such as those that address privacy concerns. Continuous context monitoring places additional emphasis on efficient resource consumption and management. Infrastructural support is essential to facilitate different context-aware applications with ease and efficiency (Kang *et al*. 2010).

Monitoring of context information can be done via a context monitoring agent such as the context manager, which will be responsible for this functionality (Pantsar-syvaniemi, Simula & Ovaska 2010). This context monitoring agent will listen or poll for changes in context information from the multiple inputs. The thresholds for the battery level is an example of the context information that the context monitoring agent will monitor for changes. The context information to be monitored is set during design time, however it can be changed at run-time by the application or the owner of the context monitoring agent.

The monitoring agent can enable subscribing in order to get a notification when the information changes or it can allow polling for the information by querying for it at certain time intervals (Pantsar-syvaniemi *et al*. 2010). Such a monitoring agent is illustrated in Figure 2.2.

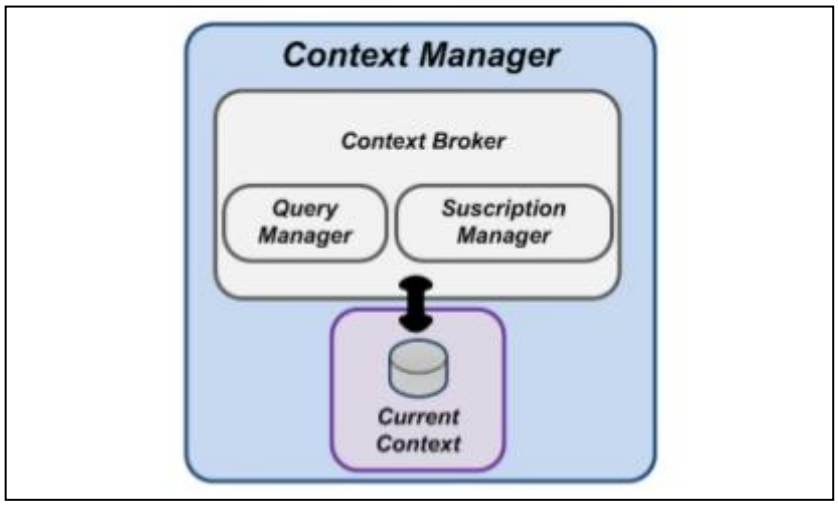

Figure 2.2: Context manager queries and subscribes to the current context (Álamo & Vilariño 2012)

Apart from monitoring context information, context information contains noise and needs to be filtered to ensure relevance and quality. The next section will highlight approaches that can assist in filtering context information.

2.2.2 Filtering Context

Context information that is gathered such as raw data after sensing is solitary, unstable and inaccurate. This context information is usually unfiltered and ambiguous, for example accelerometer data, GPS coordinates and vague text (Antila *et al*. 2011). Unfiltered context information contains a lot of noise and is not meaningful to users, especially low-level unfiltered data (Antila *et al*. 2011; Huang *et al*. 2011). Noise in context information represents random unwanted fluctuations in the measured context values. Context information that is unfiltered can also cause an error known as drift, which occurs when the actual values slowly increase or decrease from their true values over time. Another error that can occur with unfiltered data is called offset, whereby the initial sensing does not start at a zero point when it is meant to (Milette & Stroud 2012). Thus it is essential to use filters and match conditions to perform functions such as noise screening in order to effectively use the acquired sensed data.

One method to filter the errors that occur in context information is by using low-pass and high-pass filters. Even though sensors in mobile devices are continually improving, in many instances mobile applications may rely on some type of smoothing or averaging, known as low-pass filtering. Low-pass filtering passes slowly varying changes by filtering out high-frequency noise. In contrast, high-pass filtering emphasizes the higher-frequency and ignores the slow varying changes, which helps deal with offset and drift errors. Using both a low-pass and a high pass filter may be useful to highlight a specific frequency and ignore unwanted lower and higher frequencies. The use of a high-pass filter with a low-pass filter is known as a bandpass filter. Using a bandpass filter would first involve applying the high-pass filter and then the low-pass filter (Milette & Stroud 2012).

Another method to deal with the errors contained in the context information is by using Quality of Context metrics (Zheng, Wang & Kerong 2012).

Quality of Context (QoC) can be defined as:
"...any information that describes the quality of information that is used as context information. Thus, QoC refers to information and not to the process nor the hardware component that possibly provides the information" (Buchholz, Küpper & Schiffers 2003).

Quality of Context (QoC) is a measurable metric that provides information about the quality of context, which can assist in resolving uncertain and conflicting situations about context information. Context-aware applications can therefore benefit from using practical QoC metrics that are aligned with the requirements of the applications in terms of collecting, processing and provisioning of context information (Manzoor, Truong & Dustdar 2010). For example, using QoC metrics can help eliminate unwanted context data (e.g. sensor data) that does not meet the minimum quality levels. These quality levels can be explicitly set in the form of QoC thresholds or by comparing the quality of new data to previous data. The quality levels will ensure that only high-quality context information that meet the quality requirements are produced. As a result, the QoC metrics will also improve the context-aware reasoning and decision making of the context-aware application (Filho, Miron, Satoh, Gensel & Martin 2010; Manzoor *et al*. 2010).

QoC information can be implicitly gathered from mobile devices in pervasive environments. Implicitly sensing to provide context from a mobile device is a core activity in making a system context-aware (i.e. mobile application), which according to Mostefaoui, Pasquier-Rocha and Brezillon (2004), and Gray and Salber (2010) is far more complicated than explicit input to the system.

QoC information deteriorates during the process of sensing to provide context, as the QoC can be affected by the shortcomings of the sensors and the environment of a particular measurement. As a result, QoC information can be ambiguous, inaccurate and incomplete (Dey & Abowd 1999). Context-aware systems can suffer from poor performance without being able to identify the actual problem, if there is insufficient information about QoC (Manzoor *et al*. 2010). Existing context-aware applications rarely consider QoC information (Baldauf, Dustdar & Rosenberg 2007). Context-aware applications also need to make additional effort to deal with the uncertainty of context information (Ranganathan, Al-Muhtadi & Campbell 2004).

Manzoor *et al*. (2010) highlighted that existing definitions of QoC only consider it as an objective quality measure and completely disregard the multi-facetted nature of QoC in terms of being both objective and subjective. Considering both the objective and subjective natures of the QoC will assist in identifying the true quality of the context information. QoC information that is independent of the consumer's requirements is seen as the objective view of QoC. On the other hand, QoC information that is determined or derived using the context consumer's requirements is considered as the subjective view of QoC (Manzoor, Truong & Dustdar 2008; Manzoor *et al*. 2010).

The lack of context-aware applications that evaluate QoC metrics and provide them with the context information to context consumers was also emphasized by Manzoor *et al*. (2010). QoC metrics can enrich context information, which would improve the capabilities of context-aware applications to successfully use the context information to adapt to the changing situations in mobile computing environments (Manzoor *et al*. 2008).

QoC metrics can be used to identify the quality of context information from several different perspectives, such as the degree to which the context is considered fresh. QoC metrics can be measured as a decimal number with values ranging between [0..1], as quality is relative and typically matched against certain standards. A minimum value of 0 indicates that the QoC metric is in complete non-compliance to the quality requirements. A maximum value of 1, however, indicates complete compliance of the QoC metric with the quality requirements (Manzoor *et al*. 2010).

Considering the quality of the context information is an important step towards using context information effectively and achieving the capability of context awareness. Buchholz *et al*. (2003), Zheng *et al*. (2012) and Manzoor *et al*. (2010) show that the most important QoC metrics are the following:

- Freshness: Indicates validity of the context information in terms of the objective view of timeliness.
- Up-to-dateness: Indicates validity of the context information in terms of the subjective view of timeliness.
- Reliability: Indicates the extent to which context can be considered credible.
- Granularity: Indicates the precision of the context.

- Confidence Interval: Indicates the confidence in the context produced.
- Significance: Indicates the subjective importance of the context produced based on the context consumer's requirements.

Extracting simple semantic information from the sensor data and discarding the unwanted information is another vital step, which is also one of the aims of using filters (Huang *et al*. 2011). The key to extracting clear and meaningful semantic context is to develop relationships between context elements, for example the relationship between lighting conditions and temperature conditions.

Information from multiple input sources is often undesirable and sometimes even contradictory (Huang *et al*. 2011). For example, attempting to obtain the temperature from weather web services and the temperature sensor could yield unexpected and unmatched results. Context fusion enables one to maximize the effectiveness of inconsistent information from a variety of sources based on specific knowledge and rules. This knowledge and rules are needed to avoid incorrect decisions being made by the system.

Context elements that are related to or combined with other contextual elements have a greater impact and directly influence the high-level context information used by applications for the end-user. Context information whether filtered or unfiltered, needs to be stored effectively for further processing and accessibility (Huang *et al*. 2011).

2.2.3 Storing Context

Raw context data as well as filtered and combined context information can be stored in a number of places such as on a mobile device for later retrieval (Huang *et al*. 2011). This concept of making the context information persistent for the user to retrieve later is related to the passive nature of context awareness.

Context needs to be well organized into several data structures such as tables or object trees (Huang *et al*. 2011). Storing of context information is fundamental when keeping a history of context information, which will be used for further processing or needs to be accessed in the future.

The architecture for the storage of context information can be either distributed or centralized or both (Huang *et al*. 2011). For example, a distributed storage architecture could store the current context on the mobile device while storing the context history on a remote server. On the other hand, a centralized storage architecture could store both the current context and context history on a remote server or on the mobile device. The most suitable architecture for context storage is dependent on the type of application and the manner in which it will communicate with the storage mechanism. Context storage is closely linked to the modelling of context and the representation of context (Huang *et al*. 2011).

2.3 Representation of Context

2.3.1 Modelling Context

Context modelling and representation in context awareness computing is needed in order to represent a user's situation and environment (Huang *et al*. 2011). Context recognition is a process that extracts useful information from input sources (i.e. low-level context) into a representation (i.e. high-level context) that can be used by applications. Once the initial low-level context data retrieved from the input sources are pre-processed (i.e. filtered), the data can then be used in the extracting of useful information. The utilisation of thresholds-based classifiers can be used to exact the useful information. The process of performing this extraction is called feature extraction. Feature extraction is a well-known term in the area of pattern recognition. The thresholds that are used to determine extracted features provide a solution for the context recognition problem and facilitate excellent classification performance (Matyjarvi 2003). Features extracted from sensors describe context information, which are called features (Clarkson, Mase & Pentland 2000). Features can be considered a high-level view of the low-level context information obtained from the input sources (Matyjarvi 2003).

Once context recognition mechanisms are producing meaningful information, several challenges arise concerning how to represent and use contextual information (Lukowicz *et al*. 2012). Modelling each piece of context information independently is not a problem; however there is a challenge when trying to handle all the various types of context (Huang *et al*. 2011). Context modelling needs to be handled in a unified and generic context modelling manner, which is complex in today's technology conditions.

Unified modelling of context, however, can be separated into two levels. The first level is where unrelated contexts use the same data structure, such as key-value models, which would have a key and an associated value for each context. The second level can facilitate universal representation of context by means of ontology based models (Huang *et al*. 2011).

Other models such as spatial context models are considered suitable for context-aware applications that are primarily location-orientated such as mobile applications (Pantsar-syvaniemi *et al*. 2010). One thing to consider with the spatial context model is which core location model to use. Geographic location models are harder to build up than relational location models as these relational models allow simple mapping to GPS sensor data and map data. The only shortcoming is the amount of effort that the spatial model takes in order to collect and keep the latest location context information. After modelling context, contextual data is normally not in a format that can be used directly by applications and thus needs to be interpreted.

2.4 Interpreting Context

Contextual data needs to be in a usable format in order for applications to utilize it, which can be achieved via context reasoning and interpretation (Pantsar-syvaniemi *et al*. 2010). There has been an increasing availability and growing interest in ontologies (Lukowicz *et al*. 2012). These ontologies are an approach of encoding meaning and using this computationally understandable encoding to build intelligent applications. Several applications utilize open-source or application-generated ontologies (Lukowicz *et al*. 2012). W3C recommends the Web Ontology Language (OWL), which has been available since 2004.

According to Lieberman, several knowledge representation and reasoning tools and techniques are available (Lukowicz *et al*. 2012). One of these is a knowledge source developed by Lieberman called OpenMind Commonsense.

Emerging data collections such as Cycorp's online encyclopaedia, which uses knowledge representation techniques, could facilitate intelligent applications on mobile devices (Lukowicz *et al*. 2012). For example, information about the current environment could be provided to help improve location-aware applications. These data collections could also make

context-aware applications smarter by assisting them in prioritizing and filtering options shown to the user.

Context reasoning can also be used to facilitate the interpretation of context, which can be handled by a context reasoning agent (Pantsar-syvaniemi *et al*. 2010). Context reasoning involves using the received context information to justify the decision making done by an application using the contextual information. The reasoning is based on the requirements, which are normally set at design time but can also be given by the application or the user at run time. Therefore the controlling rules of the context reasoning agent are configurable. The received context information such as the high-level context information can be combined to form inferred context. An inferred context is a context that can be inferred automatically from sensors in a physical environment. Inferred context can be useful to context-aware applications (Dey, Abowd & Salber 2001).

Additional information such as global time can be used by the reasoning agents to improve the context data of the surroundings with the external data of the context space (Pantsar-syvaniemi *et al*. 2010). Even with this improved context information there are still several issues that exist when supporting context awareness.

2.5 Existing Issues

2.5.1 Sensor-based Context Recognition

Taking into account device, user and environmental information using advanced sensors and sensor networks is still a major challenge (Demeester 2010). Some researchers who used sensor-based systems have merely displayed the sensor information without actually expressing or defining what it means in terms of high-level context (Ntawanga, Calitz & Barnard 2013). For example, displaying context as separate variables, such as the current time and the current GPS coordinates, to the user. Furthermore these context variables are not discussed in terms of the integration of the different sensor information or what they mean at a higher level.

2.5.2 Activity Recognition

Determining high-level context information such as the user's current activity is a major challenge (Huang *et al*. 2011). The error-prone outputs from sensors that contain errors, such as noise, emphasises this challenge in sensor-based activity recognition (Ravi, Dandekar, Mysore & Littman 2005; Choudhury *et al*. 2008).

2.5.3 Outdoor-orientated Context Awareness

Current context-aware solutions are primarily outdoor focused and are based on location information provided by GPS or GSM networks (Demeester 2010). A possible reason for context-aware solutions only being outdoor-orientated is the difficulty of determining indoor positioning accurately.

2.5.4 Automated Situation Space Definition (Sensory Data to Situation)

Situations in context awareness are currently defined manually. This process can be challenging and subject to errors. Existing knowledge bases (e.g. ontologies of the subject area) may already have the necessary information to generate the situations and extracting the situations from knowledge bases can eliminate the need for manual work.

2.5.5 Context Prediction and Proactive Adaptation

Some papers addressed the problem of context prediction and acting on predicted context in context spaces theory. Context spaces theory is an approach to context awareness that involves using metaphors (i.e. light level - bright) for both low/high-level context. However, there are still opportunities for improvement in the field of context and situation awareness. Situations of interest at the time of system start up can be ambiguous. Identifying the areas of context that are potentially situations of interest is a topic of future work (Boytsov & Zaslavsky 2011).

2.5.6 Situation Awareness in Absence of Information

Data obtained from sensors can become invalid or missing as a result of sensors being uncertain and unreliable. The goal of situation-aware systems is to maintain as much situation awareness as possible under these conditions (Boytsov & Zaslavsky 2011).

2.5.7 Appropriate Storage of Context

Effective storage and retrieval of context information requires addressing several problems such as efficiency, reliability and relevance between context and objects, and context retrieval using ranges instead of single query conditions (Huang *et al*. 2011).

2.5.8 Balance of User Control

Lack of user control negatively affects the adoption of context-aware models and applications (Demeester 2010). This phenomenon of weak user adoption is due to the relationship between the user's expectations of the context-aware solution and the output of the context solution.

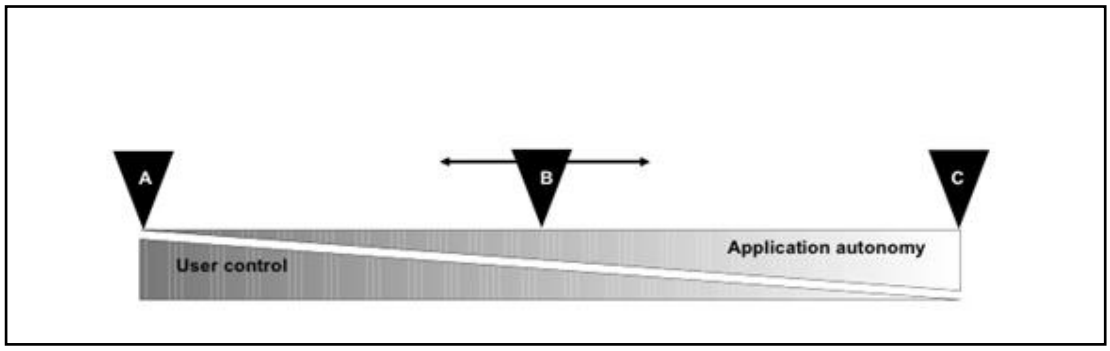

Figure 2.3: Continuum of user control versus application automation (Hardian 2011)

Context-aware solutions might not adapt as the user expects and can cause the user to experience a perceived loss of control over the behaviour of the solution. Autonomous context-aware solutions need to provide mechanisms that enable a balance between system automation and user control, as shown in Figure 2.3 (Bessi & Bruni 2009).

2.6 Possible Solutions

One approach to dealing with high-level context information such as the user's current activity is machine vision, which is focused on image processing and camera technology. Another potential approach is to check the user's calendar in order to identify what the user is meant to be doing at a specific time (Huang *et al*. 2011). Utilizing artificial intelligence techniques to recognize complex context by integrating several available low-level sensors is another promising approach. These techniques can include rules or machine learning.

Context-aware applications are generally designed using a set of if-then rules. For example, if the application senses a particular situation, then it should perform a particular action. Rules are simple to create as all the knowledge for each rule is represented in a similar format and rule-based systems are relatively simple to develop as there are several existing rule engines that determine when a rule has been satisfied. Rules are also relatively intuitive and easy to work with.

A common alternative approach is to apply machine learning. Instead of creating a sequence of rules about how an application should adapt its behaviour, an application developer can gather data on the types of situations or contexts that a user will experience and the types of desired adaptations. Machine learning can then be applied to learn the probabilistic relationships between the situations and adaptations, instead of having these relationships being hardcoded and deterministic. This still requires that the application or supporting infrastructure provides the ability to perform context inferencing to map the sensor data to user situations.

Incorporating indoor location information provided by Wi-Fi and other indoor technologies to cater for indoor context spaces such as at home or at the office can help extend context-aware applications (Demeester 2010). Using context models based on ontologies such as a Service-Oriented Context-Aware Middleware (SOCAM) can help address generating situations of interest. These models can also allow for situations to be extracted from knowledge bases without the need for manual work in defining these situations (Demeester 2010).

A context history can facilitate context predictions and pro-active adaptions by looking at the previously encountered context situations to aid future inferences of situations. Maintaining and managing a context history can address run-time situation inference when situations of interests are unknown before run-time. For example, this can be achieved by clustering context states history.

One approach to dealing with context ambiguity is to combine multiple disparate sources of the same type of context to improve the accuracy or dependability of the provided context. This is commonly known as sensor fusion (Wu 2003). For example, in activity recognition, a

Hidden Markov Model can be used with different sensors and the fused results can be represented in a table for the set of possible activities (Lester, Choudhury & Borriello 2006).

An alternate approach is to allow users to manually disambiguate ambiguity in context (Dey & Mankoff 2005). Instead of relying on an automated approach, this approach leverages a user's knowledge of the situation to help resolve and remove any ambiguity in the sensed or inferred context. A user may be presented with, for example, a list of the N most likely interpretations of context, ranked by probability and asked to select the correct interpretation. Storage of context information received from context providers can either be centralized, distributed or both. Centralized storage could utilize a server to store data such as the context history for a particular user. Distributed storage could instead store data such as user preferences and the current context on a particular user's device.

By giving users greater control over the disclosure and abstraction level of the contextual information a stronger adoption of context-aware solutions can be attained (Demeester 2010). Taking into account how the user impacts their own context, what context the user wants and what is acceptable are important questions with regards to context awareness adoption.

Table 2.1: Summary of problems matched to possible solution (Demeester 2010; Huang *et al*. 2011)

Problems	Solutions
Sensor-based Context Recognition	Multiple input sources, AI techniques and Context history
Activity Recognition	AI techniques
User Control and Automation	User Feedback
Context Ambiguity	AI techniques and User Feedback
Indoor Location Awareness	Indoor localization technologies

Table 2.1 provides a summary of the possible solutions to the existing issues when dealing with context awareness as described in Section 2.5. The overall trend in the possible solutions tend to favour AI techniques such as machine learning, use of multiple input sources such as calendar information and user feedback whereby the user can select the correct output.

2.7 Conclusion

This chapter aimed to answer Research Question 1: "*What are the context awareness problems and requirements of mobile applications?"* An explanation of context and context awareness was given. Context is essentially any information that can be used to describe the situation of an object, person or place. Context awareness refers to the idea that computers, in this case mobile devices, can both sense and react based on their environment.

Determining the context of mobile applications and how context awareness is related to location awareness and ambient intelligence was discussed. Several context dimensions were discovered, which included identity, location, status, time, device and activity. Location awareness was found to refer to devices that can passively or actively determine their location. Ambient Intelligence, on the other hand, builds upon ubiquitous computing, pervasive computing, context awareness and human-computer interaction, which is characterized by systems and technologies that are embedded, personalized, context-aware, adaptive and anticipatory.

Each of the crucial design steps to facilitate context awareness in mobile applications were discussed in detail. These steps included acquiring, monitoring, filtering, storing, representing and interpreting context.

Existing issues in context awareness were discussed, such as sensor-based context recognition, activity recognition, user control automation, context ambiguity and indoor location awareness. Possible solutions to match these issues were explored and identified. The majority of these solutions included using AI techniques, user feedback and multiple input sources. For context-aware applications to be fully supported, an underlying architecture or model is needed. This architecture or model will need to deal with the steps identified in Section 2.1 as well as address the current problems in context awareness highlighted in Section 2.5.

The next chapter will help determine the structure of the proposed model by reviewing the second DSR knowledge base identified in Section 1.4.3, which will discuss the application domain of mobile computing. Chapter 3 will discuss the need for context awareness within mobile applications. It will also discuss the medical health-care context as a potential domain

that could benefit from the proposed context-aware model and briefly review mobile applications in this domain that could benefit from facilitated context awareness. Contributions and shortcomings of existing context awareness models will be identified. Relevant context attributes and dimensions will be identified and summarized, which will assist the development of the proposed model.

Chapter 3: Mobile Solutions

3.1 Introduction

This chapter aims to answer Research Question 2: "*What are the problems and requirements of existing context awareness solutions used in mobile applications?"* This chapter also uses the Relevance and Rigor Cycle and covers the second DSR knowledge base by discussing the application domain of mobile computing. The relevance of mobile devices to support context awareness and mobile context modelling will be discussed. Context awareness in m-health as a possible sub-domain will be discussed. Existing mobile context models will be reviewed in terms of their advantages and shortcomings. The most suitable model/s that can be extended will be identified. Context attributes relevant to personal user context to facilitate context awareness in mobile applications will be identified and summarized.

3.2 Mobile Devices Supporting Context Awareness

Context information is one of the most significant information sources for mobile applications and the development of mobile devices with embedded sensors has made context data widely available (Wolf, Herrmann & Rothermel 2010). Real-time information and services anytime and anywhere has developed into a real need for people on the move. Context information such as location and calendar data, obtained from mobile phones can greatly improve the user experience in the current situation. Mobile smart phones have become the central device of users' day-to-day activities and since the context of mobile users dynamically change, so do the current needs of the users (Eichler, Lüke & Reufenheuser 2009).

Mobile smart phones incorporate a set of sensors that make it possible to capture contextual information to help individuals to better understand the surroundings that affect their daily lives (Marcu, Ghiata & Cretu 2013). The built-in sensors in modern smart phones range from GPS (for location context monitoring), image and video sensors (camera), audio sensors (microphones), light sensors, temperature sensors, direction sensors (magnetometer and gyroscope sensors), and movement sensors (accelerometers and rotation sensors) (Otebolaku & Andrade 2013).

Some of the most common sensors include the accelerometer, GPS and camera (Elgazzar et al. 2012). An accelerometer measures movement resulting from forces, including the force of gravity, applied to a device. This movement can also come from the environment wherein the device is located. The device's tilt, shake, motion, or swing can be determined from this movement data (Otebolaku & Andrade 2013).

The rotation vector sensor, which is similar to an accelerometer, is a synthetic sensor, which is most commonly used for motion detection and monitoring. The rotation vector sensor is flexible and can be used for a wide range of motion detection and related tasks such as monitoring angular change (Otebolaku & Andrade 2013).

Collecting and analysing data from sensors requires increased processing, storage and energy resources on mobile devices (Marcu *et al*. 2013). Lee, Ju, Min, Yu, and Song (2012) suggest that future developers should not only extract high-level context from raw sensing data but also make an effort to achieve optimization to support continuous sensing and processing. This optimization process is quite challenging since the resource availability of sensor devices and demands from other concurrent applications change dynamically (Lee *et al*. 2012).

Extracting of useful and meaningful high-level user contextual information from low-level smart phone sensor data has not been fully explored (Otebolaku & Andrade 2013). This gap has provided a new opportunity for mobile applications to leverage user contexts more actively, such as their location, activity, social relationship, health status (Lee *et al*. 2012).

Mobile devices are an exceptional source of context information about mobile users. Both explicit and implicit context information can be gathered from the calendar, the camera, embedded sensors or native applications (Martin, Lamsfus & Alzua 2011).

Location can usually be acquired by accessing the GPS, Cell-ID and Wi-Fi (Google Inc 2014e). Other sources of contextual information can also include information in a user's calendar or to-do list. Such information would help to develop the user's profile and could be used with other contextual information to help determine the user's context. However determining the user's context throughout their daily activities is one of the main challenges in this research area (Santos *et al*. 2009). Using multiple-input sources to identify and predict

context for given situations, such as being at home could help to solve this problem (Mitchell 2011).

Context is essential in cases such as anti-theft or near-emergency services (Santos *et al*. 2009). To provide these types of services mobile devices need to be able to clearly identify specific contexts of the user. Mobile devices with their increasing capabilities include sensors from which data such as position, lighting or sound can be obtained, which can help to determine the user's context. However, this raw context data is meaningless for computers so a suitable data model is required to represent and manage it (Martin *et al*. 2011).

An example medical healthcare (m-health) scenario for using sensor data could be to determine whether an elderly person has possibly fallen at home and has been unable to move for a period of time, consequently triggering an emergency call. However, there are multiple sources of data with unique patterns that need to be captured and processed timeously (Santos *et al*. 2009). Other challenges include the battery life of a mobile device as the energy required by context sensors is significant and can drain the battery quickly (Rahmati *et al*. 2012). M-health is a critical area in which context awareness can play an important part in assisting and providing solutions for the problems that face m-health.

3.3 Medical Context

This section introduces medical health-care as a potential domain that could benefit from the proposed context-aware model and briefly discusses several mobile applications in this domain that could benefit from facilitated context awareness. This section will help clarify the significant need that this domain has to facilitate context awareness.

3.3.1 Medical Health-care

Medical health-care can be seen as an important area for top-quality research, as it has many critical issues and problems that still need to be addressed (Bilyeu, Hardy, Katz, Kennelly & Warshawsky 2009). One of these problems is the increasing number of chronically ill patients. As a result, there is a growing burden on long-term healthcare facilities and the cost of maintaining these facilities has become unsustainable.

Patients who have chronic conditions need 24/7 monitoring. Conventional methods for monitoring of these patients typically involve the use of large, fixed equipment that restrict the mobility of the patient (Elgazzar *et al*. 2012). An important aspect of monitoring patients is the ability to track them. Locating patients can be significantly crucial, especially if they get lost as they are moved from one location to the next. Remote patient health monitoring using mobile devices can help address these constraints as mobile devices can cater for location awareness (Agarwal & Lau 2010).

Location awareness allows services to offer or access information relevant to the current situation. Patient location is an important part of contextual information that is essentially needed in healthcare systems and in particular, remote health monitoring. Location becomes a crucial context for patients who suffer from memory loss diseases such as Alzheimer's. Having access to the patient location can help to provide timely medical assistance in emergency and life-threatening situations (Elgazzar *et al*. 2012).

Understanding the context that affects health behaviour change and chronic disease management can be very important for developing effective health management practices (Klasnja & Pratt 2011). Patients can find it difficult to manage their health information if that information is in multiple locations (i.e. multiple healthcare providers) or in different formats (i.e. paper vs. electronic). A personal health record (PHR) can assist with this as it is a health record that a patient can use to store their most important health information (Health IT Inc. 2013).

An example of the data that can be contained in a PHR include (AHIMA 2013):

- Personal identification of the patient, including name and date of birth.
- Individuals to contact in case of an emergency
- Names, addresses, and phone numbers of the patient's physician, dentist, and specialists
- Health insurance information
- Living wills, advance directives, or medical power of attorney
- Organ donor authorization
- A list and dates of significant illnesses and surgical procedures
- Current medications and dosages

- Immunizations and related dates
- Allergies or sensitivities to drugs or materials, such as latex
- Important events, dates, and hereditary conditions in the patient's family history
- Results from a recent physical examination
- Opinions of specialists
- Important tests results; eye and dental records
- Correspondence between the patient and their health-care provider(s)
- Any information that the patient would like to include regarding their health such as exercise regimen or dietary preferences.

Mobile devices can make it possible to capture contextual information such as a patient's PHR to help patients and their healthcare providers. This contextual information can assist them to better understand the circumstances that affect their health and to create strategies to address those trends (Klasnja & Pratt 2011).

Mobile devices, specifically smart phones, are becoming an increasingly popular platform for the creation of health interventions (Klasnja & Pratt 2011). This phenomenon is due to the rapid developments in mobile and wireless technologies (Elgazzar *et al*. 2012).

Other factors contributing to smartphones becoming a platform for health interventions include:

- Broad adoption of mobile phones with increasingly powerful capabilities (Klasnja & Pratt 2011),
- Tendency of individuals to carry their mobile phones with them everywhere,
- Relationships that individuals build with their mobile phones, which was shown in a study conducted in Europe and India (Ventä, Isomursu, Ahtinen & Ramiah 2008),
- Context awareness features enabled through sensing and phone-based personal information.

Caregivers are showing great interest in utilizing mobile technologies that provide convenient and responsive health monitoring. A recent study conducted in the United States of America

(PricewaterhouseCoopers' Health Research Institute 2010) revealed that consumers are willing to pay for remote mobile health monitoring solutions.

It is clear that mobile devices could bring significant benefits to a vast number of individuals, such as patients who suffer from Alzheimer's disease, elderly people who are vulnerable to falling, women with high-risk pregnancies, patients who have chronic conditions that require 24/7 attention and people who need medical attention in remote and inaccessible locations or in emergency situations (Elgazzar *et al*. 2012).

With the increasing number of advancements in mobile phones such as built-in accelerometers and GPS, users' behaviours can be detected without the use of external devices (Klasnja & Pratt 2011). These technologies allow ubiquitous computing to facilitate mobility and data access anywhere and anytime (Elgazzar *et al*. 2012). User acceptance of health interventions that use mobile devices with sensor capabilities will likely increase, due to the decreasing need for users to keep track of, charge, and wear an additional device (Klasnja & Pratt 2011).

Mobile devices have become accepted as an important platform in medical health care. However, in order to fully utilise this platform and address the issues and problems in healthcare, intelligent solutions need to be built. As a result, mobile healthcare (m-health) applications could be used in order to address problems in the medical health care domain.

3.3.2 Context Awareness in M-health

Context-aware computing allows for more opportunities in application domains such as homes, offices, hotels, community areas, healthcare, campuses and military applications. These application areas are mainly intelligent spaces where users can interact with each other and conventional systems using their mobile devices (Malik, Mahmud & Javed 2007).

Smart mobile devices enable context-aware features that are significantly useful and have become an attractive platform for electronic-health (e-health) applications turning them into m-health applications (Liu *et al*. 2011).

An illustration of this is shown by Liu *et al*. (2011). Their research focused on examining from a developer's perspective, the top 200 healthcare related applications from an app store. This provided a summary of the current status and trends of iOS m-health applications and helped to identify developer implications in terms of related technology, architecture, and user interface design issues. However, only the top 100 healthcare related applications from the medical category will be discussed as they are more relevant to m-health research than the other 100 applications in the fitness category.

Liu *et al*. (2011) categorized the 100 applications into different groups according to their purposes, functions, and user satisfaction. However only 80 of those applications in the medical category were relevant to the different classes of apps.

Table 3.1: A classification of the 80 relevant applications (Liu *et al*. 2011)

Table 1
Distribution of apps among different classes of the 80 relevant apps in the Medical category.

No.	Class	Number of apps	Percentage
1	Drug or medical information database	8	10
2	Medical information reference	27	33.75
3	Decision support	3	3.75
4	Educational tools	19	23.75
5	Tracking tools	7	8.75
6	Medical calculator	3	3.75
7	Others	13	16.25

Table 3.1 shows that medical information reference was the largest class with twenty-seven applications followed by educational tools with nineteen. However, tracking tools only ranked fifth, which are generally developed for keeping track of blood pressure, diabetes factors, remote patient monitoring information and personal medical information (Liu *et al*. 2011).

Table 3.2: A classification of the 14 five-star applications (Liu *et al*. 2011)

Table 2
Distribution of apps among different classes of the 14 five-star relevant apps in the Medical category.

No.	Class	Number of apps	Percentage
1	Drug or medical information database	1	7.14
2	Medical information reference	1	7.14
3	Decision support	0	0
4	Educational tools	4	28.57
5	Tracking tools	5	35.71
6	Medical calculator	1	7.14
7	Others	2	14.29

Out of the 80 applications in Table 3.1 only 14 applications were five-star applications, as shown in Table 3.2. A five star application is an application, which was rated by users on a scale from 1 to 5 and scored an average of 5. A significant difference can be seen as tracking tools are now ranked first in Table 3.2 as opposed to fifth in Table 3.1. The cause of this trend could be interpreted as users favouring applications that could take advantage of unique features of smart phones and bring real convenience or benefits (Liu *et al*. 2011).

Context-aware systems aim to provide a context-specific service to their users by automatically adapting and staying aware of their changing contexts (Hong, Suh & Kim 2009). This is important as users with a mobile device tend to move around in their daily lives. This creates both the need and opportunity for context-aware mobile applications, which will improve services and bring added value by utilizing contextual information (Liu *et al*. 2011).

Broens *et al*. (2007) presented a framework to support the development of context-aware m-health applications. The requirements for the framework included, mobility support, context awareness and adaptation support. This context-aware framework makes the development of m-health applications feasible and improves the usability of these applications. Frameworks such as the one by Broens *et al*. (2007) enable well-designed context-aware m-health applications to be readily accepted by end users by providing mobility support, context awareness and adaptation support (Liu *et al*. 2011).

Even though few context awareness research groups focus on healthcare as a research area (Bricon-Souf & Newman 2007), it is clear that context-aware systems combined with mobile devices add significant value and could improve usability significantly (Baldauf *et al*. 2007). This was shown in a study conducted by Guerri *et al*. (2008) who developed a mobile application for assessing the muscular activity of patients with a lower back disorder. The results of the study revealed that the system was easy to use and that the users were satisfied with the usability of the application in terms of interactivity and functionality (Liu *et al*. 2011).

The results of the application survey conducted by Liu *et al*. (2011) identified several trends and important implications for app developers. One of these implications was that m-health applications that took advantage of unique smart phone features, such as context awareness, were more popular. This was validated by the trend of users giving high ratings to innovative applications such as tracking tools that took advantage of mobile device features (Liu *et al*. 2011).

3.4 Mobile Context Modelling

The current revolution of mobile devices, such as smart phones has driven a rising need for mobile services. As mobile services (applications), keep developing, the demand for context modelling of mobile users will increase (Bao, Cao, Chen, Tian & Xiong 2010).

Mobile context modelling is a process of recognizing and reasoning about contexts and situations in a mobile environment. This process is a fundamental research problem with regards to successfully leveraging the rich contextual information of mobile users while on the move (Bao *et al*. 2010).

Context information from mobile devices is primarily in the form of low-level sensor data that are not suitable for mobile applications to use (Otebolaku & Andrade 2013). In spite of vast improvements, context-aware applications still require a significantly increased context recognition accuracy for high-level context information based on imprecise sensor data to enable the robust execution of context-aware applications (Wolf *et al*. 2010).

Robust context models can help solve this problem as they capture the relevant contextual information to be considered in context-aware applications. This contextual information can be categorized into different dimensions, such as social relationships, location, time and environments. These dimensions can be seen as the fundamental primitive elements in a context model (Chang, Barrenechea & Alencar 2010).

New applications and usage scenarios have developed and will develop, such as social community services including Facebook and LinkedIn. These applications and scenarios will require a scalable, extensible context provisioning framework, e.g. new context types/domains and evolving context models. However building new context-aware applications is still complex and lengthy due to the lack of a suitable model or middleware-level support (Fausto & Alberto 2010).

A suitable model for context-aware applications needs to be well established and should provide support for several tasks dealing with context. These tasks include performing context interpretation and dealing with acquiring context from various sources such as device sensors, databases and web services (Fausto & Alberto 2010). In order to manage and use context information it must be represented and stored in some form that can be used by machines to derive higher-level context, as well as communicated and modelled. Context is just a special type of metadata and so it is open to all the techniques used in meta-modelling.

Since high-level context is derived from relationships between more primitive contexts, storing context in a relational database and querying the database to derive this higher-level context is a common approach (Fausto & Alberto 2010). Using unsupervised artificial intelligence learning techniques is also another approach, which has the ability to learn personalized contexts of mobile users, which are difficult to predefine (Bao *et al*. 2010).

Bao *et al*. (2010) suggested that incorporating domain knowledge for common contexts, such as "having dinner" with unsupervised approaches for mobile context modelling could be future work. This semi-supervised approach has the potential to improve the learning performances of common contexts while keeping the flexibility of supervised approaches for learning personalized contexts.

Existing context models, however, do not effectively deal with dynamic aspects of contextual information such as location, time, social relationships, and changing preferences. These current models lack a suitable design or focus on modelling static aspects of context, as user context is by its very nature highly dynamic (Chang *et al*. 2010).

3.5 Existing Mobile Context Models and Infrastructures

Research on context-aware systems has focused on many different areas. This research ranges from application-level implementations to frameworks and context models, and its applications cover various domains. Several models and infrastructures have been developed, which each focus on their own set of attributes and dimensions. Each of these will be discussed below in terms of their overall completeness to fully support the personal user context of a mobile user.

Gehlen, Aijaz, Sajjad, and Walke (2007) introduced a context dissemination middleware based on a mobile Web Services framework. Their mobile context dissemination middleware focused mainly on context attributes including, location, time, task, network, user context and social circumstance. They did not incorporate a health context, which could be seen as part of user context and instead only included a health scenario where their model could be useful.

Falchuk, Loeb, and Panagos, (2008) focused on the challenges of creating middleware that offers rich context-aware event logic to address a spectrum of issues across many verticals. The context attributes that they addressed include:

- Personal information (e.g. interests, expertise)
- Social information (e.g. contacts, relationship, medical)
- Subscriptions (e.g. traffic alerts, various feeds)
- Device states
- Presence and availability (e.g. willingness to communicate)
- Privacy and access control

- Schedule (e.g. calendar, to-do lists).

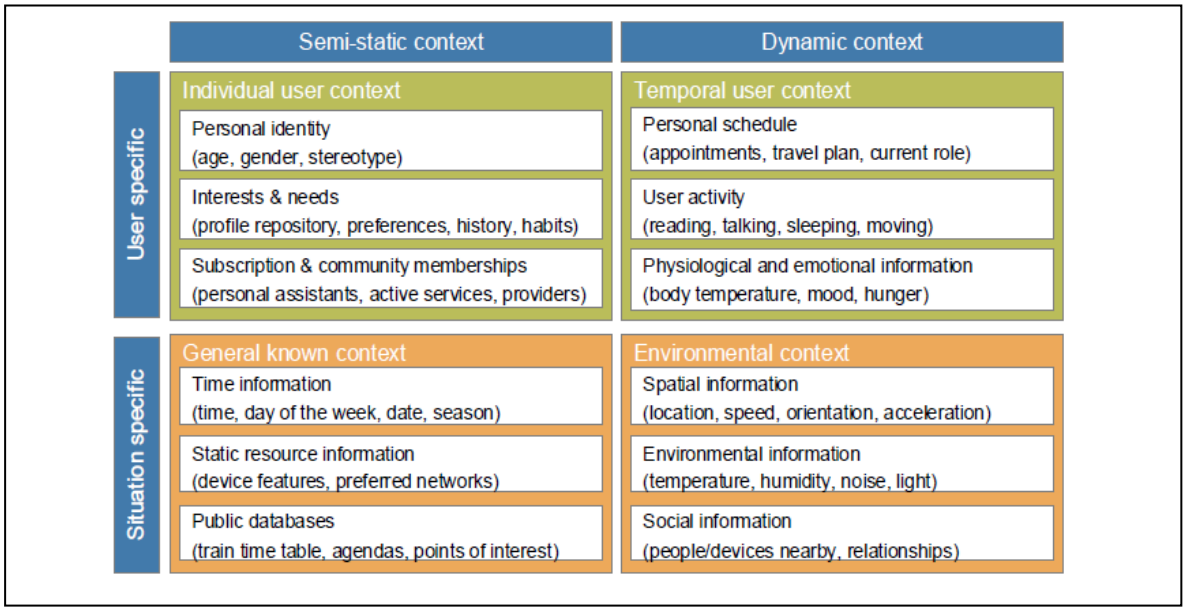

Figure 3.1: Classification of user and situation context parameters (Eichler *et al.* 2009)

Eichler *et al.*, (2009) addressed an approach to service offering and usage on mobile phones. They focused on a typical scenario in public transport and classified their context parameters into semi-static and dynamic contexts, as well as into user and situation specific, as shown in Figure 3.1.

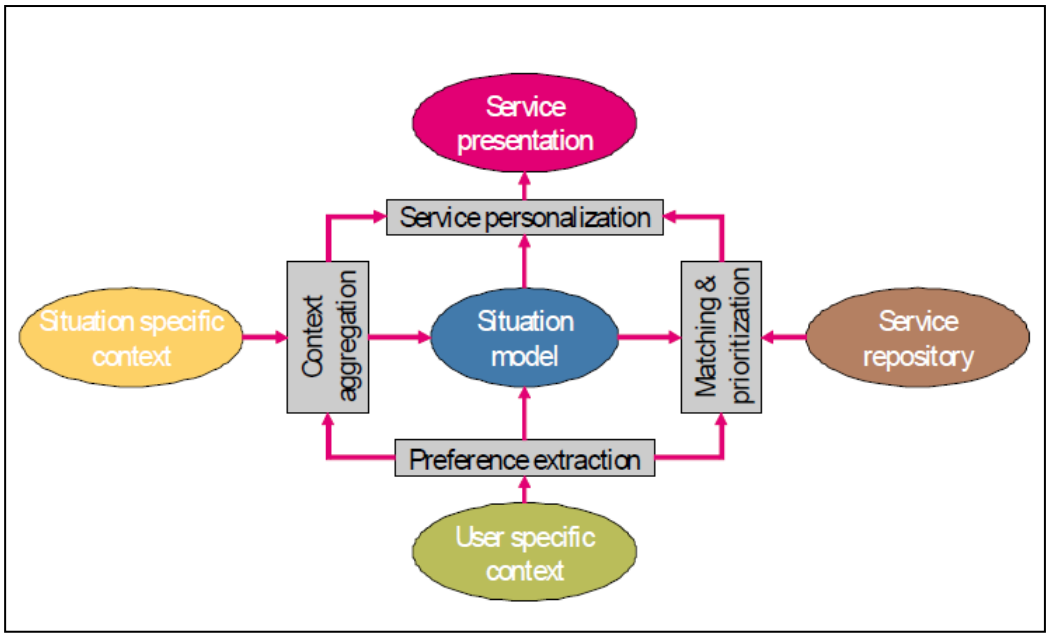

Figure 3.2: Context-sensitive composition framework (Eichler *et al.* 2009)

Eichler *et al.*'s (2009) situation model highlighted the importance of separating situation specific and user specific contexts to support the right services being provided at the right time based on the current context, as shown in Figure 3.2. Other than the aggregated situation context, the extracted user preferences serve as additional input for an improved adaption of the service offering to the actual requirements of the user. However, the user specific context of the model does not include a health context parameter, which can be seen as a part of a mobile user's personal user context.

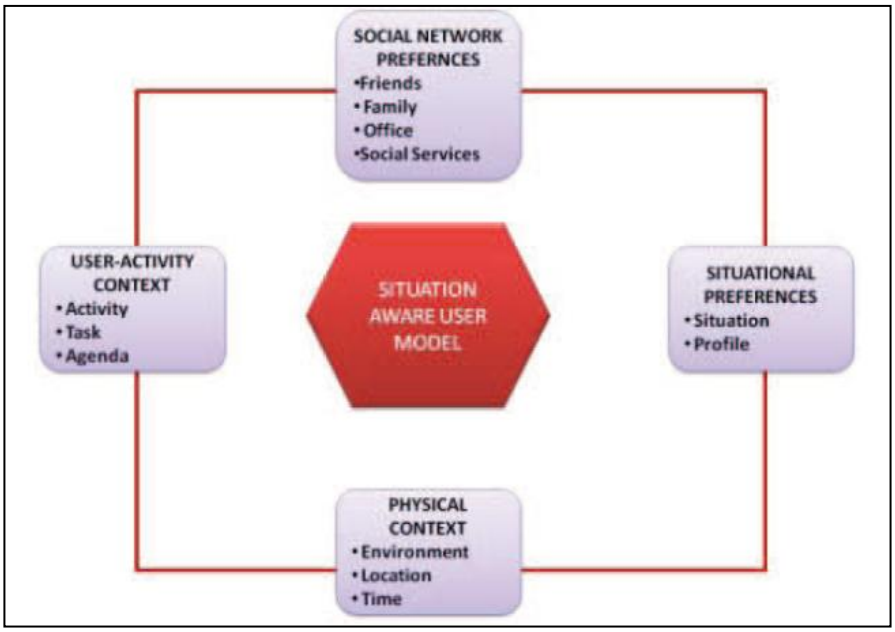

Figure 3.3: Situation-aware user model (Fausto & Alberto 2010)

Fausto and Alberto (2010) developed a situation-aware user model to make context-aware mobile services possible and for these services to adapt to changing contexts and user needs. The situation-aware user model sets preferences for the user for a given context and has elements such as social network preferences, situational preferences, physical context and user-activity context, as depicted in Figure 3.3.

Their model suggested that by including social network information as part of context, the user's personal context, metadata of information and recommendations provided to users could be enhanced. This situation-aware user model focused on more context dimensions than Eichler *et al.*'s (2009) situation model. However, it also does not include the health context of the user.

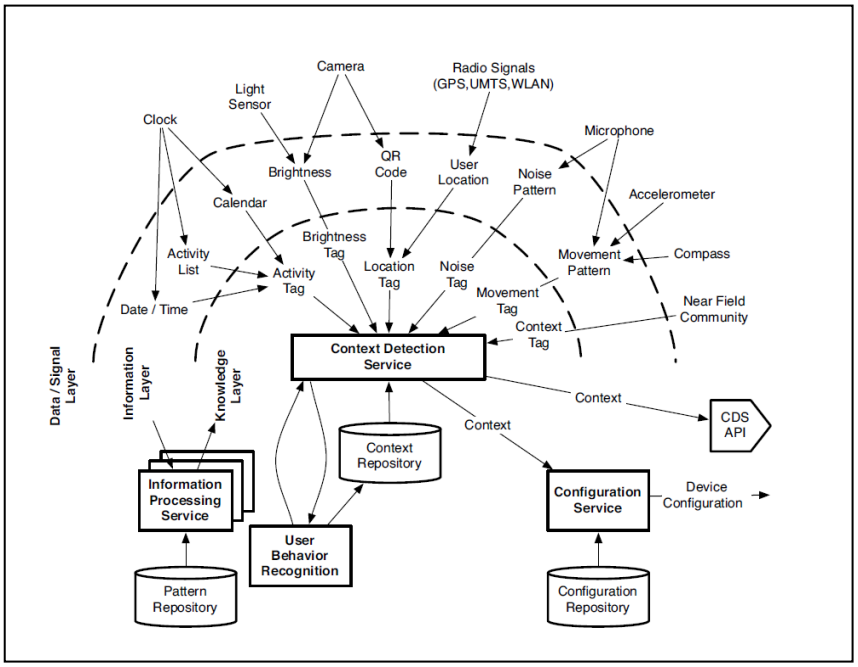

Figure 3.4: Context detection service architecture (Christoph, Krempels, Stulpnagel & Terwelp 2010)

Christoph et al. (2010) introduced an integrated approach for the automatic detection of a user's context. Their Context Detection Service Architecture as shown in Figure 3.4, was however, primarily sensor and behaviour-based and made no reference to a personal user profile or user and situation preferences.

The Context Detection Service Architecture proposes that detection of a mobile user's context should be provided by the mobile device as a service. This would act as an API, so that all applications can have access to the context information and adapt their behaviour accordingly.

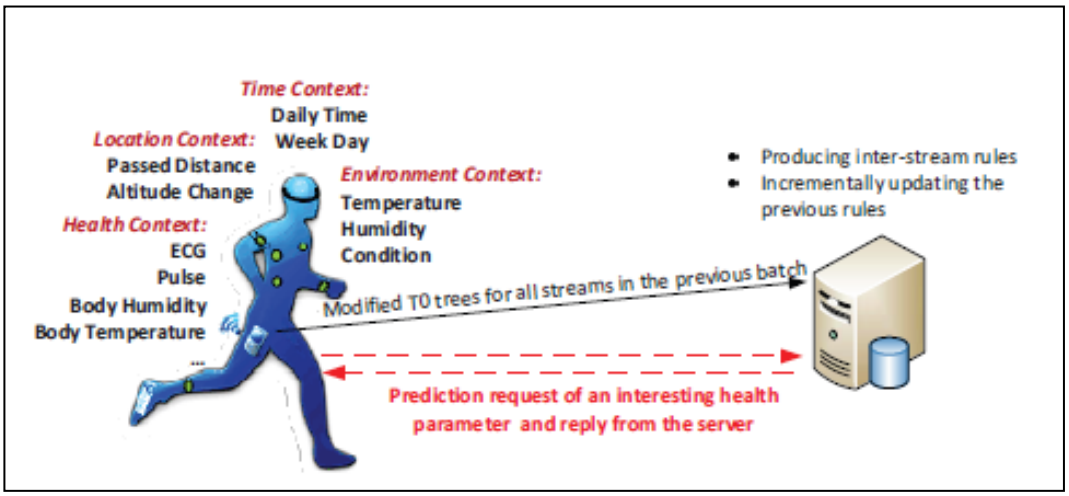

Figure 3.5: The mobile health context prediction scenario (Hassani & Seidl 2011)

Hassani and Seidl (2011) introduced one of the first methods for predicting an added health context of a mobile user. Unfortunately, the model used required users not only to have a mobile device but also to be equipped with body sensors, which is impractical and could lead to poor user acceptance.

Alidin and Crestani (2012) utilized a "just-in-time" approach, in which the relevant information is retrieved without the user requesting it. They provided more details in terms of how context could be identified and captured but their infrastructure does not support the use of non-sensor data such as calendar and preferences.

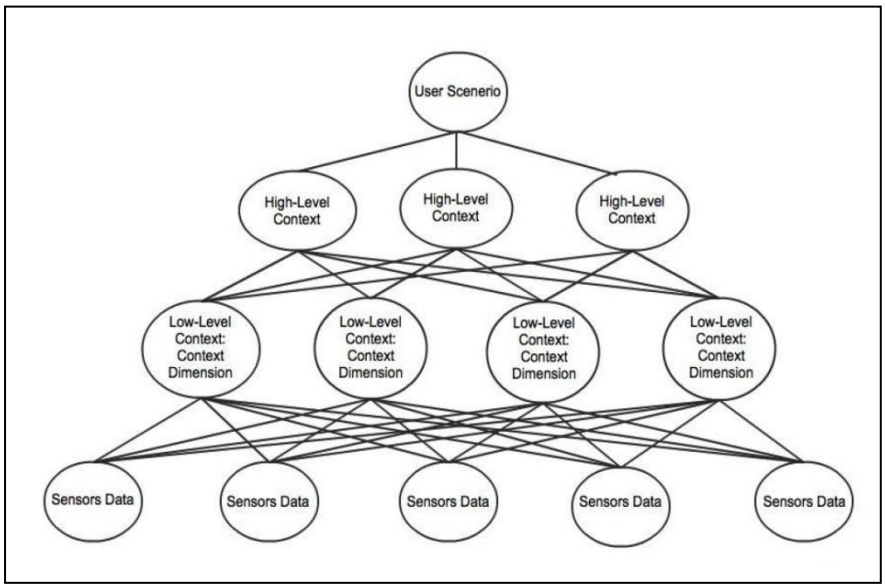

Figure 3.6: Four levels interpretation of context model (Alidin & Crestani 2012)

They developed their own context interpretation model in order to translate sensor data into the description of user context. This model consists of four different levels of context interpretation, as depicted in Figure 3.6. This model, however, is too high-level and is missing other user specific context data sources such as user profiles and preferences.

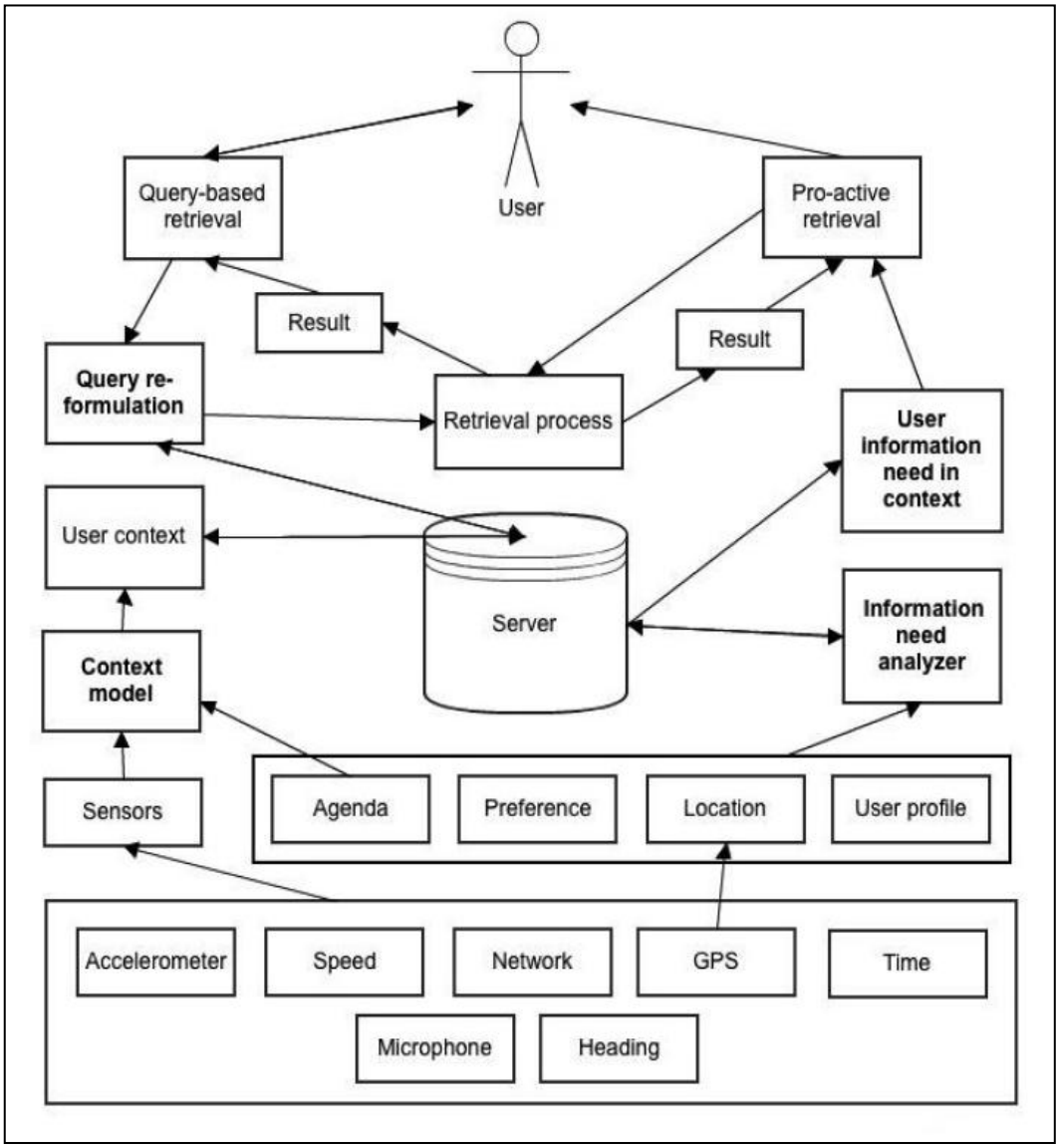

Figure 3.7: JIT-MobIR conceptual architecture (Alidin & Crestani 2012)

Figure 3.7 helps to clarify the missing data sources in their context interpretation model and separates sensor data from user data. It unfortunately does not incorporate a health context as part of the user data. This architecture does highlight different context retrieval processes that could be used in order to retrieve the user's context, the first being a query-based retrieval and the second a pro-active retrieval.

Figure 3.8 explores the context dimensions that Alidin and Crestani (2012) focused on, which highlight the fact that context attributes also comprise of several states. For example, the location context has three states including: home, workplace and unknown.

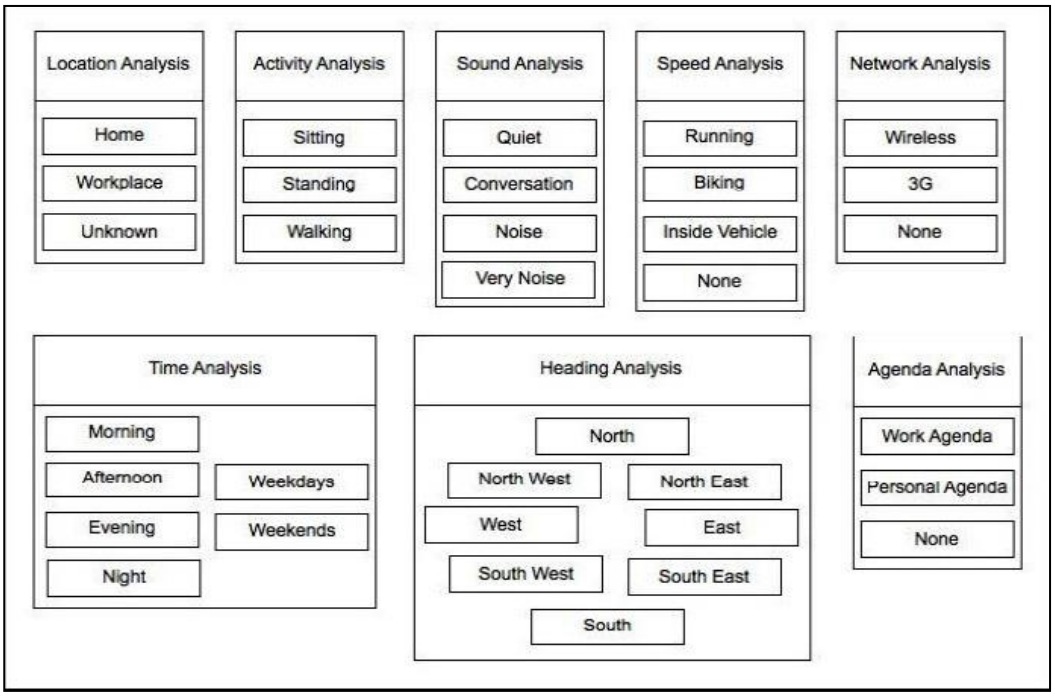

Figure 3.8: General sensors analysis for the context dimensions (Alidin & Crestani 2012)

Reviewing the existing context models and additional literature highlighted that there are many context dimensions and attributes. However, not all dimensions and attributes are seen as relevant in terms of personal mobile user context. Key trends in existing models include the use of preferences as well as separating context information regarding the physical environment from information about the user. From the existing models that were reviewed, the situation-aware model proposed by Fausto and Alberto (2010) appears to be the most complete in comparison to the other models. However, this model does lack several components including a health context dimension as well as context attributes such as device states.

3.6 Context Dimensions and Attributes

The context dimensions identified in Chapter 2, including identity, location, status (user perceived properties such as environmental), time, device and activity, were used as a basis for the building blocks (i.e. dimensions and attributes) of the proposed model. Building upon these dimensions with previous literature in this chapter, a combination of the most relevant and common dimensions and attributes were synthesized. A summary of the most relevant dimensions and examples of associated attributes appears in Table 3.3.

Table 3.3: A summary of relevant context attributes and dimensions for personal user context

Context Dimensions	Context Attributes	Input Source	Available
Location	Current location, destination	GPS, Time, Calendar, IPS, Wi-Fi	Yes
Time	Time, day of week, date	Mobile device	Yes
Activity	Walking, running, driving	Sensors	Yes
Schedule	Appointments, task list, travel plan	Calendar, To-do list	Sometimes
Physiological	Body temperature, mood, hunger	Body Sensors	No
Identity	Age, gender, stereotype	User input	Yes
Interest & needs	Preferences, history, habits, profile repository	User input, Sensors	Yes
Points of interest	Locations of interest	Location, Interests, Sensors	Sometimes
Social	Contacts, relationships	Mobile device	Sometimes
Device States	Features, Sensors	Mobile device	Yes
Availability	Willingness to communicate	User input, Sensors	Sometimes
Environmental	Temperature, noise, light, humidity, forecast	Sensors	Yes
Spatial	Speed, orientation, acceleration	Accelerometer, compass, gyroscopes.	Yes
Network	Selected network, available networks	Wi-Fi, 3G	Yes
Subscriptions	Traffic alerts, various feeds	Web services	Sometimes
Health	Conditions, Blood type, Allergies	Personal Health Records (PHRs), user input	Sometimes

3.7 Conclusion

Mobile smart phones incorporate several sensors that make it possible to capture contextual information to help individuals to better understand the surroundings that affect their daily lives. The most important sensors among these instruments are the accelerometer, GPS and

camera. Extracting useful and meaningful high-level user contextual information from low-level smart phone sensor data has not been fully explored. This research gap has provided a new opportunity for mobile applications to leverage user contexts more actively, such as the users' location, activity, social relationship and health status. Some researchers suggest that future developers should not only extract high-level context from raw sensing data but also make an effort to optimise the context solution to support continuous sensing and processing.

Medical health-care can be seen as an important area for research, as it has many critical issues and problems that still need to be addressed. Patients who have chronic conditions need 24/7 monitoring. An important aspect of monitoring patients is the ability to track them. Locating patients can be crucial, especially if they get lost as they move from one location to the next.

Understanding the context that affects health behaviour change and chronic disease management can be very important for developing effective health management practices. Mobile devices can make it possible to capture contextual information such as PHRs to help patients and their healthcare providers. This contextual information will assist them to better understand the circumstances that affect their health and to create strategies to address those trends. Mobile devices, specifically smart phones are becoming an increasingly popular platform for the creation of health interventions. This phenomenon is due to the rapid developments in mobile and wireless technologies.

The results of the application survey conducted by Liu *et al*. (2011) identified several trends and important implications for application developers. One of these implications was that m-health apps that took advantage of unique smart phone features, such as context awareness, were more popular. This is validated by the trend whereby users gave high ratings to these new, innovative apps such as tracking tools that took advantage of these mobile device features (Liu *et al*. 2011).

Mobile context modelling is a fundamental research problem with regards to successfully leveraging the rich contextual information of mobile users whilst on the move. Context-aware apps still require a significantly increased context recognition accuracy for high-level context information on inaccurate and imprecise sensor data to enable the robust execution of context-aware apps. Robust context models that capture contextual information can be

categorized into different dimensions, such as location, time and environments. These dimensions can be seen as the fundamental primitive elements in a context model.

Suggestions that incorporate domain knowledge for common contexts, such as "having dinner" with unsupervised approaches for mobile context modelling could be topics for future research. This semi-supervised approach has the potential to improve the learning performances of common contexts while keeping the flexibility of supervised approaches for learning personalized contexts. Existing context models do not effectively deal with dynamic aspects of contextual information such as location, time, social relationships and changing preferences. These current models lack a suitable design or focus on modelling static aspects of context, but user context is by its very nature highly dynamic.

Eichler *et al*.'s (2009) situation model highlighted the importance of separating the user and situation specific contexts. It also had a well-structured classification of the context attributes utilized within the situation model. Their model did not, however, include a health context. The situation model by Fausto and Alberto, (2010), also highlighted the separation of user and situation specific contexts, but more importantly it emphasizes the need to have a set of preferences. Alidin and Crestani's (2012) model on the other hand provides an overall picture of context interpretation but lacks specific details in terms of other sources of context including user profile and preference information. The underlying conceptual architecture associated with Alidin and Crestani's (2012) model provides a useful structure whereby sensor data is separated from user data.

These models and infrastructures were considered the top three of those reviewed, with the most complete model being the situation-aware model proposed by Fausto and Alberto (2010). Their model contained the key aspects highlighted from the other two models including separation of situation/physical from user context and an overall picture of the different context dimensions involved. However, their model still lacked several key aspects to successfully model personal user context of a mobile user. Fausto and Alberto's (2010) model together with other additional elements such as an added health context and the context attributes (Table 3.3) can be used as the basis of a new and improved mobile context-aware model.

This chapter aimed to answer Research Question 2: "*What are the problems and requirements of existing context awareness solutions used in mobile applications?"* The relevance of mobile devices to support context awareness and mobile context modelling were discussed. Existing mobile context models were reviewed and suitable model/s that can be extended were identified. Context attributes relevant to personal user context to facilitate context awareness in mobile applications were also identified and summarized.

The next chapter will address Research Question 3: "*How can an improved context-aware model be developed?"* It will also discuss the design and implementation of the proposed context model for a personal user, which will use multiple input sources.

Chapter 4: Design and Implementation

4.1 Introduction

This chapter addresses Research Question 3 that was identified in Chapter 1: "*How can an improved context-aware model be developed?"* The main aim of this chapter is to discuss the design and implementation of the context-aware model. This design and implementation falls into the Design Cycle of the DSR methodology. This Design Cycle allows for generated feedback to be used to iteratively refine the design of the context-aware model and the development of the prototype. The typical context awareness architecture (Figure 2.1) discussed in Chapter 2 can be improved to support context awareness in mobile applications using multiple input sources. The most suitable existing context model that was selected in Chapter 3 will be used as the basis for the proposed model and will be extended to address its shortcomings. The context attributes that were also identified in Chapter 3 that are relevant to personal user context will be matched against the existing context model and its context attributes from each context dimension. The contents of this chapter were incorporated in a paper presented at the Southern Africa Telecommunications Networks and Applications Conference in 2014 (Pather, Wesson & Cowley 2014). The feasibility of the artefact is an important aspect of DSR (Hevner *et al*. 2004).

The next section of this chapter will discuss the design of the proposed context-aware model. Each of the components that form part of the proposed model will be discussed. The improvements incorporated in each component will also be highlighted during the discussion. The implementation of the context-aware model will follow the design section. The implementation section will address each of the processes required to implement the proposed model. The tools and techniques used for each of the processes will also be discussed. The main deliverables of this cycle are the design of the context-aware model and the implementation of this model in the form of a prototype, named CoPro. The prototype was named CoPro as the prototype is meant to act as a context provider. The prototype will be developed in an iterative manner, whereby changes will be made based on constant feedback as the prototype is implemented.

4.2 Design

The requirements and outcomes from Chapters 2 and 3 play an important role in the design of the proposed context-aware model. To ensure that the proposed model can support context awareness in mobile applications using multiple input sources these requirements and outcomes need to be addressed. This section highlights how each of the components that form part of the proposed model should be designed to support context awareness in mobile applications.

4.2.1 Context Attributes

One of the outcomes from Chapter 2 is that several crucial design steps are required to facilitate context awareness in mobile applications. The first of these steps involves acquiring the context of a mobile application. Context data can be gathered from multiple input sources, such as sensors, the calendar and web services. Context data can be broken down into context attributes and context dimensions. Context attributes are the building blocks of context dimensions. Context dimensions include a set of context attributes that define the specific context dimension. For example, the model depicted in Figure 4.1 has a Physical Context dimension with context attributes, which are environment, location, time, spatial, network and device states. Several context attributes for personal user context exist in related literature. The most frequent context attributes were synthesized and summarized, as depicted in Table 4.1.

These summarized context attributes are listed with the multiple input sources required to gather data for that attribute. For example, in order to obtain an accurate user location, inputs from the GPS, calendar and web services can be used. Each attribute was also given example values. The example values can be classified as actual or fuzzy, depending on the values. If the raw data is sufficient, the value type will be actual, or if a semantic meaning can be used to better represent the raw values, then the value type will be classified as fuzzy. For example the attribute time can have a value of 04:47 PM, which would be classified as an actual value. This value can be further processed to obtain a more high-level context meaning by applying a threshold to derive a fuzzy value of Afternoon based on the absolute value of 04:47 PM. The application of thresholds to these context attributes will be discussed in detail in the section which deals with the feature extraction process of implementation.

Table 4.1: Summarized context attributes for personal user context

Attribute	Input Source	Example Value
Environment	Ambient temperature sensor, sound sensor, light sensor, pressure sensor, web services.	32 degrees, cloudy, 120 lux
Location	GPS, time, calendar, IPS, Wi-Fi, web services.	lat 35 long 24/ Summerstrand
Time & Date	Mobile device	04:47 PM/ Afternoon/ 14 March 2014/ Friday
Spatial	Accelerometer, linear acceleration, orientation sensor, gyroscope, rotation vector and GPS.	34km/h, North-East
Network	Wi-Fi, 3G	Wi-Fi, 3G/ NMMU student Wi-Fi, MTN
Device States	Mobile device	Battery level, charging state
Activity	Accelerometer, linear acceleration.	Driving, still, on foot.
Task	To-do list	Bake cake
Schedule	Calendar	Masters Meeting, Cake day
Identity	Facebook, user input (profile)	Male, Tom, 7 Jan
Social	Facebook, mobile device	Richard, Friend
Physiological	Body sensors	64 degrees
Medical	User input (profile), PHRs	Type 2 Diabetes, A+, Hay fever
Situation	Mobile device	Silent, Loud
Interest & Needs	User input (preferences), Facebook	The Beatles, Pizza
Points of Interest	User input (preferences)	KFC, Boardwalk
Availability	Calendar, accelerometer, linear acceleration.	Available, Busy

All of the above-mentioned context attributes are considered to be part of the personal user context of a mobile user. This research focuses on supporting context awareness in mobile applications by only using the capabilities and resources accessible on a single smart phone mobile device. Thus some of these attributes will not be implemented in the prototype, but will still be included in the proposed model. For example the attribute Physiological, can currently only be obtained via body sensors. This attribute is seen as an input source that would require more than one device to capture and thus will be excluded during the implementation of the proposed model. The remaining context attributes will be used in the design of the proposed model.

4.2.2 Proposed Model

The proposed model was designed based on the synthesized context attributes illustrated in Table 4.1 and contained in the Situation-Aware User Model identified in Chapter 3 as the most suitable to be extended. The synthesized context attributes were matched to the context attributes of the original model and the remaining context attributes that were not in the original model were categorized and added to a related dimension.

This proposed model extended the original model to include a health context with physiological and medical as context attributes. The context attributes spatial, network and device states were also added to the Physical Context. All of the preferences of the original model were combined into one preference dimension, which included a social attribute. The preferences were combined in order to consolidate all the preferences into one dimension, which could then be managed by the preference manager in the adapted architecture shown in Section 4.2.3. Finally the remaining context attributes were also added to preferences, which included interests and needs, points of interest and availability. The situation attribute was classified as a preference in the existing model, however, as the ringer state forms part of the device's state it was moved to device states in the Physical context. The proposed model is depicted in Figure 4.1 with the changes to the original model highlighted in light blue for new context dimensions and in dark red for the new context attributes.

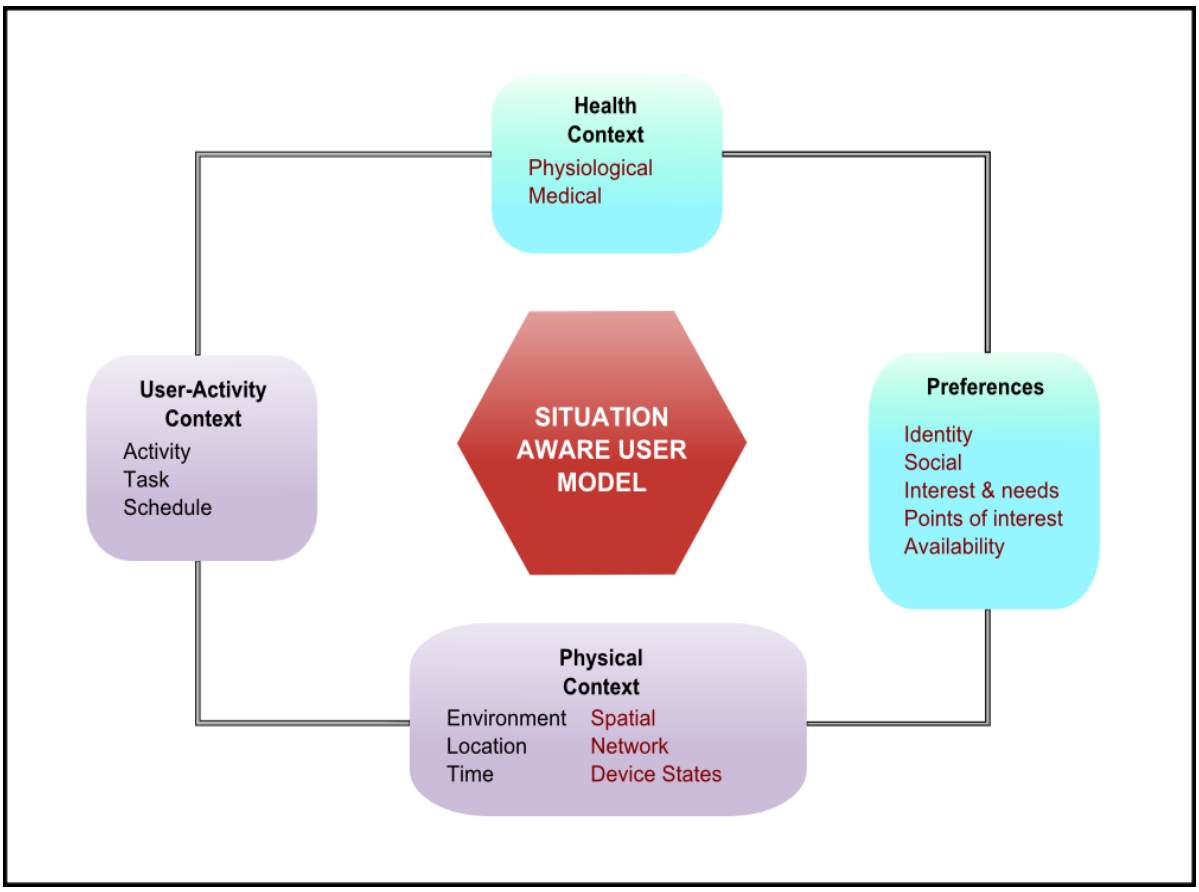

Figure 4.1: Proposed model with four core dimensions

The proposed model consists of four main context dimensions: Physical, User-activity, Health and Preferences. These four dimensions are described in more detail below to indicate what context information they cover.

The *Physical Context* dimension has six context attributes, namely:

- Environmental - Light level, temperature level, humidity level, proximity level sound level.
- Location - Current location.
- Time/Date - Time of day, day of the week.
- Spatial - Device orientation, movement speed.
- Network - Network status.
- Device States - Battery level, charging state, ringer state.

The *User-Activity Context* dimension has three context attributes, specifically:

- Activity - Movement activity.
- Task - General goals.
- Schedule - Current calendar event.

The *Health Context* dimension has two context attributes, namely:

- Physiological - Bodily measurements.
- Medical - Personal Health Record (PHR).

The *Preferences* dimension has five context attributes:

- Identity - User profile.
- Social - Friends list.
- Interests & needs - Books, music, movies.
- Points of interest - Places of interest.
- Availability - User availability.

The proposed model provides a comprehensive approach to modelling personal user context. The proposed model alone, however, is not sufficient to support context awareness and thus the underlying typical architecture discussed in Chapter 2 needs to be adapted to suit the proposed model.

4.2.3 Adapted Architecture

The typical architecture identified in Chapter 2 (Figure 2.1) has several aspects, such as the middleware layer with a context and a preference manager, that are necessary to support the proposed context model. This architecture, however, does not cater for multiple input sources and does not clearly illustrate the different context dimensions supported by the proposed model. Thus this architecture was adapted by adding additional input sources including calendar, GPS/IPS and web services, to the gathering and pre-processing layer as shown in Figure 4.2. These inputs, together with the sensor inputs, will be processed and combined to form context dimensions, which will represent outputs from the gathering and pre-processing layer. The middleware layer has the typical context manager with a context model but also

has the preferences dimension, which was added to the preference manager. The red lettering used in Figure 4.2 highlights the additional changes that were made to the original architecture (Figure 2.1). The core of the research was focused on the middleware, and context gathering and pre-processing layers as these layers form the foundation for supporting context awareness. The context-aware applications layer including the programming toolkit is part of future work in which the implemented prototype would be converted into an Application Programming Interface (API) for mobile developers to use in their mobile applications.

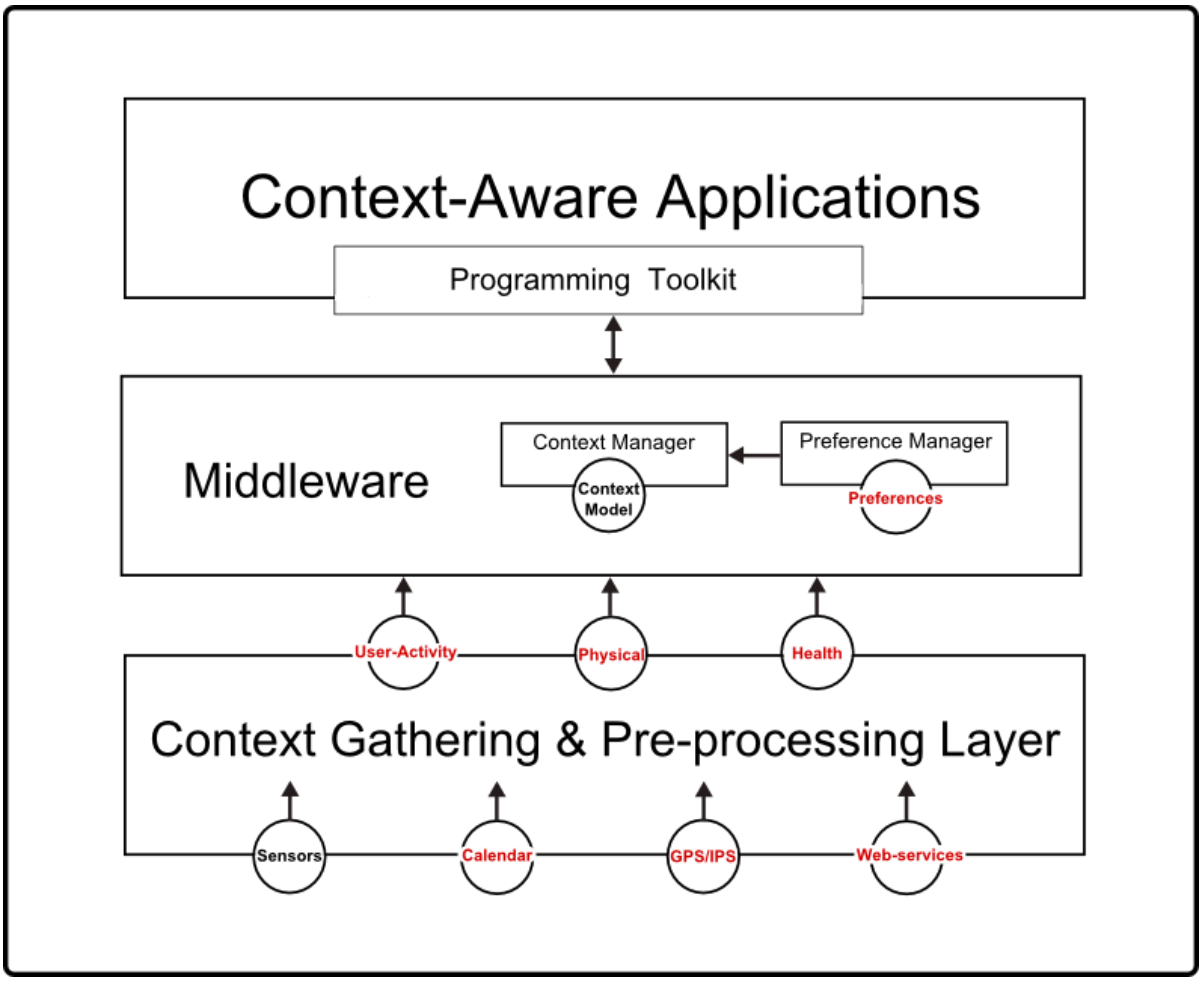

Figure 4.2: Adapted architecture with multiple-input sources and context dimensions

The adapted architecture consists of three main layers: context gathering and processing, middleware and context-aware applications. These three layers will be highlighted in more detail below to indicate the context processes that they handle.

The Context Gathering and Processing layer is responsible for:

- Gathering the raw context data from the multiple input sources that are accessible only from a single mobile device. These include sensors, calendar, GPS/IPS and web services.
- Pre-processing the raw context data by using filters to filter out noise and drift, Quality of Context (QoC) metrics to evaluate the quality of the data and applying thresholds to derive a higher-level abstract meaning.
- Categorizing the processed context data into three dimensions, namely physical, user-activity and health.
- Handing over the processed context data within the three dimensions as inputs to the middleware layer.

The Middleware layer is responsible for:

- Using the processed context data from the context gathering and pre-processing layer and the preferences from the preference manager as inputs.
- Feeding the processed dimensions and context data into the context-aware model and using the preferences to tailor the outputs.
- Finally providing the inferred context outputs to the context-aware applications layer.

The Context-Aware Applications layer is responsible for:

- Interacting with the middleware layer via a programming toolkit such as an API in order to retrieve the inferred context data outputs.
- Use the inferred context data to improve the user experience of the mobile application that uses the context outputs.

The adapted architecture better supports the proposed model as it now caters for multiple input sources and also supports the four context dimensions of the proposed model. This research is primarily focused on the context gathering and pre-processing layer and the middleware layer. However, in order for the context-aware applications layer to communicate effectively with the middleware layer, a data design is needed. This data design will provide

an easy access point to the context-aware applications attempting to obtain inferred context data.

4.2.4 Data Design

Using the proposed model, the data design was formulated as an UML class diagram, as illustrated in Figure 4.3. As personal user context is focused on the context of a person, the data design was structured around a Person class. This Person class is associated with each dimension class with a multiplicity of 1..1. Each dimension class will be accessible from the Person class. For example, to access the data from the Physical Context class, a method person.getPhysicalContext() will be called. The person.getCurrentContext() will provide access to all of the dimension classes with a single call. The User-Activity and Health Context Classes also includes a date and time attribute in the data model as these two classes are also dependent on date and time.

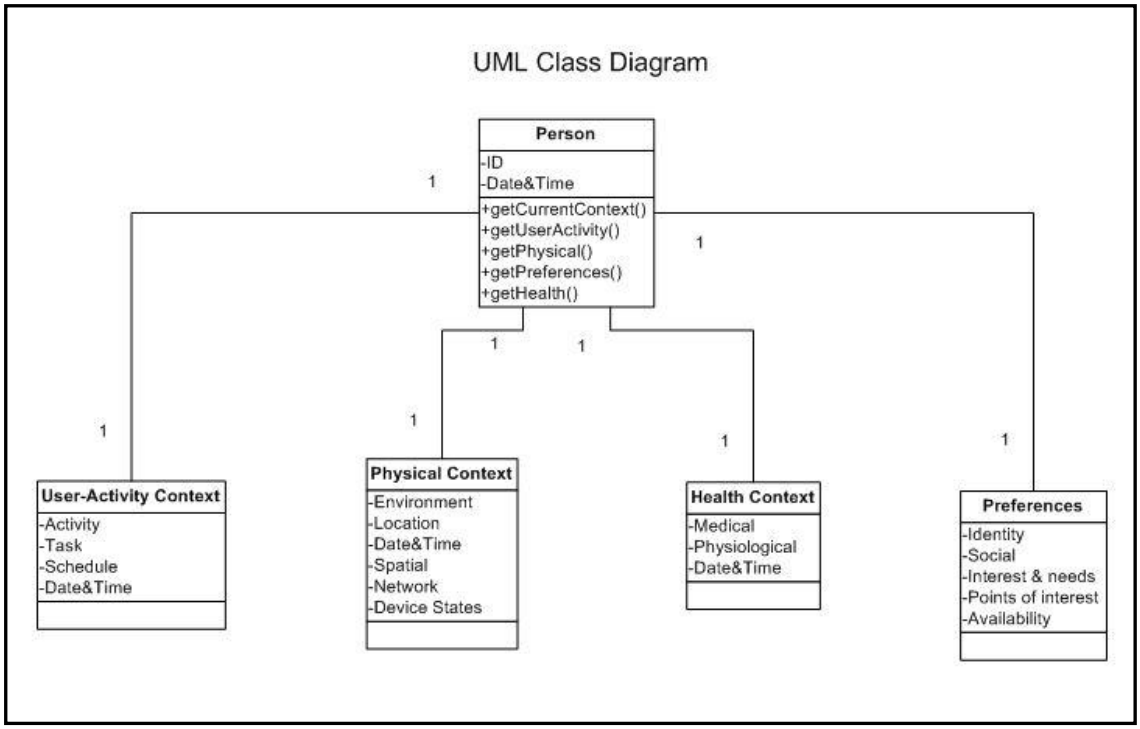

Figure 4.3: UML class diagram data design

The UML Class Diagram provides a logical data design that will enable the context-aware applications to easily access the inferred context data provided by the middleware through an API.

4.3 Implementation

The previous section highlighted the design of the proposed model and the supporting underlying architecture. The design of the proposed model was discussed, which was based on a set of context attributes, a situation-aware user model and a typical architecture to support context-aware applications. This section discusses the implementation of the proposed model by describing how the design of the proposed model was implemented as a prototype, named CoPro, the main screen of which is shown in Figure 4.4. The main screen highlights the key functionality provided by CoPro, which includes:

- Gather Data - Gathers and displays raw data from multiple inputs.
- Available Sensors - Displays list of available sensors on the device.
- Quality of Context - Calculates and displays QoC metrics.
- Feature Extraction - Generates and displays features from the gathered data.
- Infer Context - Generates and displays inferred context from features.
- Preferences - Displays user preferences including health preferences.

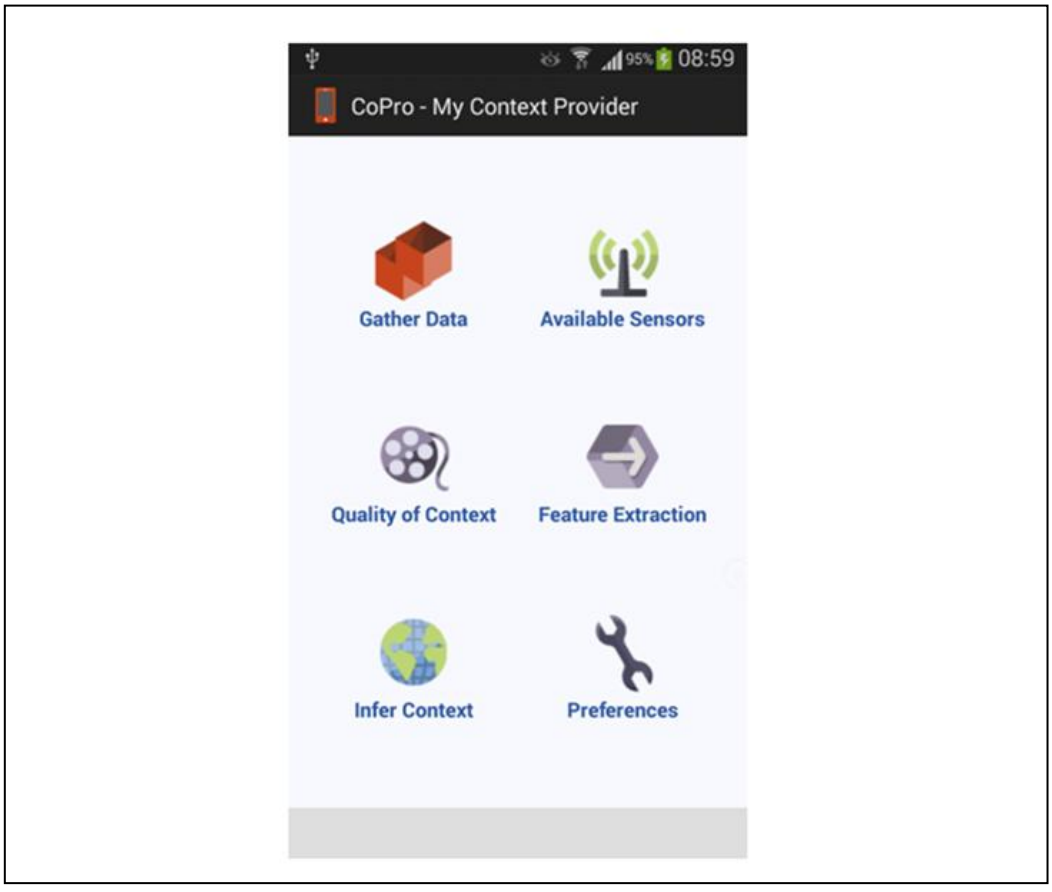

Figure 4.4: Main screen of CoPro prototype

CoPro was developed to handle all the complex processes required to support context awareness that were highlighted in Section 2.1.1 in Chapter 2. These complex processes involved as well as the tools and techniques used during the implementation will be discussed in detail.

The implementation of the prototype based on the proposed model involved the following steps:

- Gathering of context data from multiple-input sources.
- Filtering of the gathered context data with QoC metrics (Section 2.2.2).
- Defining and applying thresholds to the filtered data for feature extraction.
- Collecting and training of processed context data to produce context models.

The implementation of the prototype followed an iterative development process following the rigor cycle of the DSR methodology. Following this iterative development process made the development of the prototype more manageable as each complex process was reliant on the previous process functioning properly. The physical context dimension was implemented first, followed by the user-activity context, the health context and finally the preferences context dimension. The prototype was developed on a Samsung Galaxy S4, which had an Android operating system of Version 4.1.2 (Jelly Bean) (Google Inc. 2014c). The Samsung Galaxy S4 was selected as it contains a large number of built-in sensors that are applicable to this research. These sensors included a variety of new sensors that were not previously available on the Samsung Galaxy S3, such as the ambient temperature and humidity sensors. After the mobile device and targeted platform were selected, the next step was to start gathering the context data.

4.3.1 Data Gathering

The gathering of context data involves two of the significant steps identified in Chapter 2, namely acquisition and monitoring of context data. Context acquisition is focused on gathering an instance of the current context data, while monitoring of context deals with checking for changes in the context data. For example, when dealing with location data, acquiring the user's location provides a single instance of the location data. However, if wanting to monitor the user's location as his/her location changes, one has to actively monitor it by requesting to receive periodic location updates. To receive periodic location updates two

update parameters need to be specified. The first parameter is the update interval, which is set in milliseconds. Once set, the update interval is not guaranteed as it is dependent on other location services requests, which are competing for time slices on the smart phone. If there are no other requests the periodic location updates will be received at the preferred update interval that was set (Google Inc. 2014i). The second parameter determines the accuracy required for the location request. If a location request of high-accuracy is needed the priority is set to high accuracy. The choice in update parameters can significantly affect the battery life of the mobile device and thus needs to be handled accordingly (Google Inc. 2014e).

The data gathered for the four contexts were obtained from multiple input sources. The integrated approach discussed in Chapter 2, of using sensory data from the mobile device and the network in the form of web services, was followed. The multiple input sources used are highlighted below for each dimension according to the design of the proposed model. The web services used as input sources will also be discussed and motivated for each of the web services selected.

The *Physical Context* dimension used the following input sources:

- Environmental - Light sensor, temperature and humidity sensor, proximity sensor, microphone, web services (OpenWeatherMap).
- Location - GPS, Wi-Fi, GSM, sensors, web services (Google APIs).
- Time/Date - Mobile device.
- Spatial - Accelerometer, gravity sensor, magnetometer, gyroscope, rotation vector sensor, GPS.
- Network - Mobile device.
- Device States - Mobile device.

The *User-Activity Context* dimension utilized the following input sources:

- Activity - Low-power sensors (Google API).
- Task - To-do list. (Out of scope)
- Schedule - Calendar on mobile device.

The *Health Context* dimension used the following input sources:

- Physiological - External wearable sensors. (Out of scope)
- Medical - Personal Health Record (PHR).

The *Preferences* dimension utilized the following input sources:

- Identity - web service (Facebook).
- Social - web service (Facebook).
- Interests & needs - web service (Facebook).
- Points of interest - Places of interest. (Out of scope)
- Availability - Calendar, activity.

As mentioned before, the physiological context attribute will not be implemented as it is out of scope. The additional context attributes that will also not be implemented include tasks and points of interest context attributes. The task attribute cannot be implemented at this stage as there is currently no generic to-do list that is found on mobile devices such as the calendar. For this reason it was considered out of scope. The points of interest attribute requires input directly from the user as highlighted in Table 4.1 in Section 4.2.1. Since there is no other identified input source for this attribute, it has also been classified as out of scope.

The main input sources that were used during the data gathering process consisted of the input from the built-in sensors and elements within the mobile device such as the date and time. The other most significant input sources included the web services and APIs used. Web services are a type of API, which usually operates over HTTP, however, not all APIs are web services (Cholakian 2009).

The most suitable web services and APIs were selected based on the input data required and the output data returned. The selected web services and APIs include:

- OpenWeatherMap API for obtaining weather data;
- Google Geocoding API for determining addresses;
- Google Location APIs (Fused Location provider and Activity Recognition);
- Facebook API for obtaining most of the preference data.

In order to handle the battery life of the mobile device more efficiently, several data gathering strategies were implemented in CoPro. The first strategy implemented involved combining movement awareness with location awareness. If the mobile device has not yet moved, then the location would still be the same. However, if the device has moved, the location would have changed. This initial strategy was achieved by using the values of the activity recognition API in determining when to request new location updates from the fused location provider API. For example, if an activity of value of "Driving" was detected with a high confidence (i.e. 100%), then the request for new location updates with a high-accuracy were made. On the other hand, if an activity of value of "Still 100%" was detected, no further location updates were made. Alternatively, if an activity of value of "Unknown" was detected, new location updates with a balanced-accuracy (i.e. balance between power and accuracy) were made. This initial strategy not only preserved battery life, but also made effective use of movement awareness with location awareness in CoPro (Google Inc. 2014e).

The second strategy implemented involved creating a dependency between the request for weather data and request for location updates. This dependency was identified as significant as the request of current weather data is reliant on a single location (OpenWeatherMap Inc. 2014). This second strategy was achieved by only requesting weather data when an updated location was received. The implementation of this second strategy was enhanced further by only requesting a weather update if the weather data had not yet been set or if the weather data was older than ten minutes (i.e. 600,000 milliseconds). The age of the weather data was captured in milliseconds, highlighting the importance of this unit of measure.

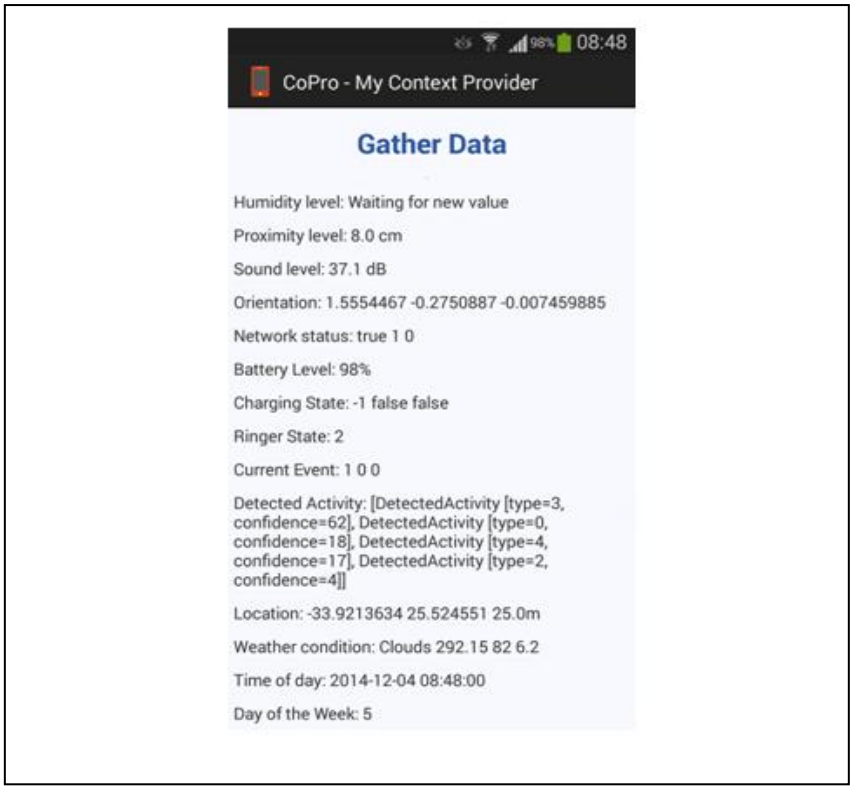

Figure 4.5: Example of low-level data gathered by CoPro

Once all the data was gathered by CoPro, as shown in Figure 4.5, some of this data needed to be filtered in order to be more useful. An example would be the sensor data that is acquired from the built-in sensors of the mobile device, which are regarded as raw sensor data. In most cases raw sensor data needs to be pre-processed or filtered in some way to cater for any reading errors. There are several tools and techniques available that can overcome these errors. The selected techniques used during the implementation of the proposed model will be highlighted in the next section.

4.3.2 Data Filtering

The most suitable tools and techniques were selected and used based on the gathered context data. These tools and techniques were also selected based on the errors that they deal with and the amount of processing required for the raw data to become useful as the processing was done on the mobile device.

Initially, a bandpass filter was selected to filter the motion data (i.e. accelerometer data) as it consists of a high-pass filter as well as a low-pass filter (Milette & Stroud 2012). The high-pass filter would filter out slow drift (i.e. slow increasing or decreasing of values) and offset

(i.e. initial reading not starting at 0) while giving higher frequency changes. The low-pass filter would have a smoothing effect (i.e. averaging) on the data and filter out high-frequency signals or noise. The bandpass filter would therefore filter out both low-frequency and high-frequency data and keep the data at a frequency range with fewer errors. However, a bandpass filter was not used as the rotation vector sensor, as identified in Section 3.2 in Chapter 3, was used for implementing motion-related tasks such as monitoring angular change. By using the rotation vector, no further calibration or filtering was needed. This deduction was supported by the rotation vector sensor itself, which is an integration of accelerometer, gyroscope and magnetometer readings. The combination of these different sensor readings allowed for the errors from one sensor to be mitigated by the other. For example the rotation vector uses a gyroscope and an accelerometer as its main orientation change inputs. The gyroscope, however, is not perfect as it has a drift error and thus a magnetometer input is needed to compensate for this gyroscope error. Drift describes the slow, long-term wandering of returned sensor data away from the real-world value (Milette & Stroud 2012).

The microphone was not part of the other sensors and thus needed to be handled and processed independently. For example, the sound values obtained from the mobile device's microphone was not only filtered by using a simple moving average (i.e. mean of every 10 values reported), but also had to be calibrated. This calibration involved calibrating the sound values with a sound level meter in order to accurately report the actual ambient sound level. This calibration was needed to report valid sound level values as the mobile device's microphone did not report the actual sound level.

The raw environmental data collected from the light, pressure and temperature sensors usually require no calibration, filtering or modification (Google Inc. 2014a). Other raw data from sensors such as the proximity sensor also typically require no filtering. However, when assessing the accuracy attribute provided with the environmental and proximity sensor readings, a value of 0 was returned, which is also represented as "SENSOR_STATUS_UNRELIABLE". This value indicated that some calibration was needed (Google Inc. 2014h). In order to address this issue and to produce optimal results, a fixed buffer window was used.

After gathering data from each sensor, a sensor data object for each reading was created to handle the multiple values obtained from the sensor readings. For example, the light sensor reported several values, including light level, accuracy, type of sensor and timestamp (Google Inc. 2014g).

The timestamp of each sensor reading did not produce accurate data regarding when that specific sensor reading was actually measured. This is a known issue regarding the timestamp information provided from mobile device sensors (Google Inc. 2014b). In order to solve this problem, the following programming method System.currentTimeMillis() was used to capture the current time at the time a specific sensor reading was measured (Sam 2012). This captured time was used as the measured time of that specific sensor reading and was then used to create the sensor data object together with the other reported values (i.e. light level, accuracy and type of sensor). This sensor data object was then added to the fixed buffer.

The size of each buffer varied depending on the type of sensor and the frequency with which the sensor reported new sensor readings. For example, the light sensor has a minimum delay (i.e. minDelay) of 0, which indicates that it receives a new reading as soon as a change in the light level is detected. The occurrences of light sensor readings were tested during development, which reported between seven to eight sensor readings per second. Based on this number of readings per second, the size of the buffer was set to eight for the light sensor. This buffer size ensured that enough light sensor data objects were collected for further processing.

To further process the sensor data objects in the fixed buffers, a set of QoC metrics as identified in Section 2.2.2 in Chapter 2, were used to evaluate the quality of the data in order to filter the values. The objective of filtering the values in the buffer, was to determine the best sensor data object by assessing the quality of those objects gathered with QoC metrics. The best sensor object was determined by first taking every sensor object out the buffer and creating a new context data object (e.g. a light data object). On the creation of the second context data object, the QoC metrics for both context objects were calculated with those QoC values then being compared. The values from the QoC metrics are between 0 and 1, as highlighted in Section 2.2.2, making them easier to compare. A calculate and compare process was repeated for all sensor objects in the buffer, whilst temporarily storing the

current best context data object of the two. Using the light sensor as an example, the implemented filter process is highlighted in the pseudo code shown as Figure 4.6.

```
If Sensor = Light Sensor
  {
   Then for each sensor object in fixed buffer
    {
      if no value for current best object
      {
         current best object = create new light context object
                                (using sensor object)
      }
      else
      {
          next object = create new light context object
                         (using next sensor object)

          Calculate all QoC metrics for current object
          Calculate all QoC metrics for next object

          Compare QoC values of current with next
          result1  = 1st QoC value of current minus 1st QoC value of next
          result2  = 2nd.....
          result3  = etc..... for all QoC metrics

          Sum = result1 + result2 + result3
          if Sum less than 0
           {
             then current best object = next object
           }
      }
   }
}
```

Figure 4.6: Pseudo-code representing QoC filter process

As illustrated in Figure 4.6, each QoC result (e.g. result1) was determined by subtracting the next object's QoC value from the current object's QoC value. Each result was greater than 0 if the current object's QoC value was higher than the next object's QoC value or if vice versa, then the result would be less than 0. These QoC results were then summed and if that overall summed value was less than 0 then the current best context data object would be the next object. If the overall summed value was greater than 0, the current best object remained the same and was then compared with the next context object created from the buffer.

All the QoC metrics used in this filter process as well as those implemented for assessing the quality of other non-sensor inputs (e.g. network state, calendar, weather) are discussed below, highlighting how each metric was calculated. The calculated values of the QoC metrics were also stored in each context data object. The QoC metrics implemented as identified in Section 2.2.2 in Chapter 2, included freshness, up-to-dateness, reliability, granularity, confidence interval and significance

The first QoC metric was freshness, which was a quality measure that was used to calculate the objective timeliness of each context object. The freshness metric was considered objective as it was measured independent of specific user requirements (Manzoor *et al*. 2010). The freshness QoC metric was calculated using the ratio of age of context object O and the time period, which was determined by using Equation 1.

$$\text{Age(O)} = t_{curr} - t_{meas}(\mathcal{O}) \quad (1)$$

The age of context, as shown above, was calculated by subtracting the measured time t_{meas} of context object O from the current time t_{curr}. The age of context highlighted how old the context object was based on the current time.

The second part needed to calculate the freshness QoC metric was the time period. The time period as defined by Manzoor *et al*. (2010) was the time interval between sensor reading measurements. This definition, however, only compensates for sensor inputs and not all context inputs. This deduction was further supported by Manzoor *et al*. (2010) who suggested using the sensor configuration (i.e. minDelay) as the time period. The minDelay value, however, is only provided by sensors and not provided by other context inputs, such as calendar and network state. Another issue with the minDelay, is that it only reports the minimum time that the sensor can produce new sensor reading measurements but not the actual time between readings (Google Inc. 2014f). Manzoor *et al*. (2014) later reworded their time period definition as the time interval between two readings of context. Nonetheless, this reworded definition still only referred to sensors as it was classified under sensor characteristics. By taking this reworded definition of time period into consideration, a possible solution was derived for calculating the time period for all context inputs. The solution for calculating the new time period is highlighted by Figure 4.7 shown in Code Extract 1.

```
(1)   Line 1 - long prevTime = curr_Network.cMeasuredTime.getTimeInMillis();
      Line 2 - String ct = sdf.format(new Date(System.currentTimeMillis()));
      Line 3 - curr_Network.updateTime(ct);
      Line 4 - long currTime = curr_Network.cMeasuredTime.getTimeInMillis();
      Line 5 - long timeP = currTime - prevTime;
      Line 6 - curr_Network.timePeriod = timeP;
```

Figure 4.7: Code extract of code used to calculate time period for network state

The code extract in Figure 4.7 is an example of how the new time period was calculated for the non-sensor network state context, which would be used for calculating the network's freshness. Line 1 shows the code used to keep a reference to the measured time of the previous network context object. The current time was then obtained and assigned to a variable (i.e. ct) in Line 2. Line 3 updated the measured time for the previous network object with the new measured time (i.e. ct). Line 4 shows the code used to keep a reference to the updated measured time of the current network object (i.e. updated with ct). The time period was then calculated by subtracting the measured time of the previous network object from the measured time of the current network object in Line 5. Finally the current network object's time period was set in Line 6.

The freshness QoC metric was then finally calculated for each input context object O with Equation 2:

$$\text{Freshness(O)} = \begin{cases} 1 - \frac{Age(\mathcal{O})}{TimePeriod(\mathcal{O})} & : \quad if \quad Age(\mathcal{O}) < TimePeriod(\mathcal{O}) \\ 0 & : \quad otherwise \end{cases} \qquad (2)$$

As shown in Equation 2, if the age of context object O was less than the time period of context object O, freshness was then calculated by the normalized (i.e. subtracted from 1) ratio of age to time period. However, if the age of context object O was not less than the time period of context object O, the freshness was reported as 0.

The up-to-dateness was the second QoC metric used, which measured the subjective timeliness of each context object. In contrast to the freshness metric, up-to-dateness was

considered subjective as it was measured based on specific user requirements (Manzoor *et al*. 2010). The only difference between calculating the up-to-dateness and freshness was the use of a subjective validity time value instead of an objective time period value. This validity time value was set for each individual input source based on the maximum time interval in which the input source was considered stable (Manzoor *et al*. 2010). The up-to-dateness was calculated for each input context object O using Equation 3:

$$\text{Up-to-dateness(O)} = \begin{cases} 1 - \frac{Age(\mathcal{O})}{ValidityTime(\mathcal{O})} & : \quad if \quad Age(\mathcal{O}) < ValidityTime(\mathcal{O}) \\ 0 & : \quad otherwise \end{cases} \tag{3}$$

The structure of the up-to-dateness formula as shown in Equation 3, was similar to that of the freshness formula as shown in Equation 2 with the only exception being the use of the validity time value as a substitute for the time period value.

The third QoC metric used was reliability. Reliability measured the reliance in the correctness of the context information. Manzoor *et al*. (2014) identified and calculated reliability by using the accuracy provided by the sensor, and the distance between the context source and the user for which that information was collected. However, there were two issues regarding their calculation of reliability for this research.

The first issue was that the accuracy value provided by the sensors was not reliable as identified at the beginning of this section (Section 4.3.2). The second issue was that the reliability calculation considered context sources (i.e. sensors) that were remotely located sensors, such as temperature sensors distributed around a city. As this research only focused on the input sources (e.g. sensors) provided on the mobile device, the distance between the context source and user was negligible. Therefore a new method to calculate the reliability for each input context object O was produced and is shown in Equation 4:

$$\text{Reliability(O)} = (\text{Freshness(O)} + \text{Up-to-dateness(O)}) / 2 \tag{4}$$

The produced reliability formula as shown in Equation 4, used the mean of the freshness and up-to-dateness QoC metrics of context object O in order to determine the reliability of

context object O. Using the freshness and up-to-dateness metrics together provided both an objective and subjective view of the reliability of context object O.

Granularity of a context object measures the precision of the context data in terms of the maximum precision level that the context object can obtain (Zheng *et al*. 2012). For example, a sound value of 32.22 dB would have a granularity of 1 if the maximum precision level for sound was 3. The maximum precision value of sound of 3 in this example was calculated by adding 1 to the maximum number of decimals a sound value could contain (i.e. 2). Granularity was only calculated for contexts with multiple levels of precision, which included the location context as well as the streaming sensors consisting of light, temperature, humidity and sound context. Each context identified, had a unique maximum precision value, which was dependent on the highest precision level that context could produce. The granularity for each input context object O of the identified contexts was calculated using Equation 5:

$$\text{Granularity(O)} = \frac{Current\ \text{Pr}ecisionLevel}{Numberof\ \text{Pr}ecisionLevel} \tag{5}$$

Granularity of context object O was calculated by using the ratio of the current precision level to the number of precision levels (i.e. maximum). The current precision was calculated at runtime when the granularity of context object O was calculated.

The next QoC metric used was the confidence interval of a context object. Although the confidence interval was not a typical QoC metric, it was still used as one. This was because the confidence interval provided valuable insight into the quality of the context information produced, which is the purpose of a QoC metric (Section 2.2.2). The confidence interval was only obtained for the location and activity context as the context providers of these input sources provided a confidence interval with the context data (Google Inc 2014d). Therefore no calculation of the confidence interval was needed.

The final QoC metric used during the implementation was the significance QoC metric. The significance QoC metric measured the extent to which the context information was considered important for a specific situation (Zheng *et al*. 2012). The significance of a context object was determined by considering the critical value of the context object in

relation to a maximum critical value of that context. The significance QoC metric is based on user requirements and was classified as a subjective QoC metric (Manzoor *et al*. 2010). The significance of context object O was determined by means of Equation 6.

$$\text{Significance(O)} = \frac{CV(\mathcal{O})}{CV_{max}(\mathcal{O})} \quad (6)$$

As illustrated in Equation 6, the significance of context object O was determined by the ratio of the current critical value of the context object O to the maximum critical value of context object O (Manzoor *et al*. 2010).

All the above QoC metrics formed the basis of the evaluation metrics used for the evaluation of the prototype, CoPro, which was developed based on the proposed model. The use of QoC metrics as evaluation metrics will be discussed in the next chapter, Chapter 5.

Once the best context data object was obtained based on the QoC metric filtering, the data object was then prepared for feature extraction, as identified in Chapter 2 (Section 2.3). This process was used to extract high-level context from the low-level context, which is described in the next section.

4.3.3 Feature Extraction

Determining the user's context throughout their daily activities is one of the main challenges in this research area, as identified in Chapter 3. Extracting useful and meaningful high-level contextual information about the user from raw low-level data of a mobile device (i.e. smart phone) is a step towards solving this problem.

Feature extraction is achieved by setting thresholds for the processed data, which allow for a high-level meaning to be obtained rather than simply reporting the absolute value. For example, the light sensor reports its readings in lux and has a dynamic range between 0 and 30,000 lux. The smallest difference in light that the sensor can detect is 1 lux. A value of 10,000 lux is regarded as an overcast day (i.e. high-level). The following numbers represent typical values that can be expected (Milette & Stroud 2012):

- No moon - 0.001 lux

- Full moon - 0.25 lux
- Cloudy - 100 lux
- Sunrise - 400 lux
- Overcast - 10000 lux
- Shade - 20000 lux
- Sunlight - 110000 lux
- Sunlight max - 120000 lux

The high-level meanings used for multiple input sources gathered are shown below for each dimension according to the design of the proposed model. The feature extraction was only applied to the dynamic data and not the static data, as the dynamic data has multiple states as opposed to the static data that only has one state (i.e. actual value). Only the physical and the user-activity dimensions had dynamic data and therefore feature extraction was only applied to these two dimensions to obtain high-level meanings. The high-level meanings that were used were based on the data collected as well as existing thresholds (Santos, Cardoso, Ferreira, Diniz & Chaínho 2010; OpenWeatherMap Inc 2014; Google Inc 2014d).

The *Physical Context* dimension used the following high-level meanings (Santos *et al*. 2010; OpenWeatherMap Inc 2014):

- Environmental
    - Light level - Very Dark, Dark, Normal, Bright, Very Bright
    - Temperature level - Very Cold, Cold, Mild, Hot, Very Hot
    - Humidity - Low, Medium, High
    - Proximity level - Near, Far
    - Sound level - Very Silent, Silent, Moderate, Loud, Very Loud
    - Weather Conditions - Overcast clouds, Clear sky, Scattered clouds, Broken clouds, Shower rain, Rain, Thunderstorm, Snow, Mist.
- Location
    - Current Location - Actual Address
- Time/Date
    - Time of day - Dawn, Morning, Afternoon, Night
    - Day of the week - Monday, Tuesday, Wednesday, Thursday, Friday, Saturday, Sunday.

- Spatial
    - Orientation level - Face up, Face down
- Network
    - Network Status - Wi-Fi & Mobile Enabled, Wi-Fi enabled, Mobile Enabled, No Internet access
- Device States
    - Battery level - Low, Okay, High, Full
    - Charging State - Charging with USB, Charging with AC, Discharging
    - Ringer State - Silent, Vibrate, Normal

The *User-Activity Context* dimension utilized the following high-level meanings (Google Inc 2014d):

- Activity
    - Detected Activity - Unknown, Still, Tilting, on Foot, Cycling, Driving
- Schedule
    - Current Event - Actual Event

Extracting features from the multiple-input sources helped to provide more meaningful high-level information such as providing the location address instead of reporting only the GPS coordinates. This provided insight into the user's current context rather than simply reporting absolute values and addressed the first existing issue identified in Section 2.5 in Chapter 2. The high-level context that was produced for CoPro as a result of the feature extraction process is highlighted in Figure 4.8. The feature extraction process (i.e. settings thresholds) was one step towards addressing the problem of determining the user's context throughout their daily activities. The next step, as suggested in Section 3.2 in Chapter 3, was to not only identify high-level context from multiple-input sources (i.e. low-level), but to predict context for specified situations, such as being at home, which could help to solve this problem.

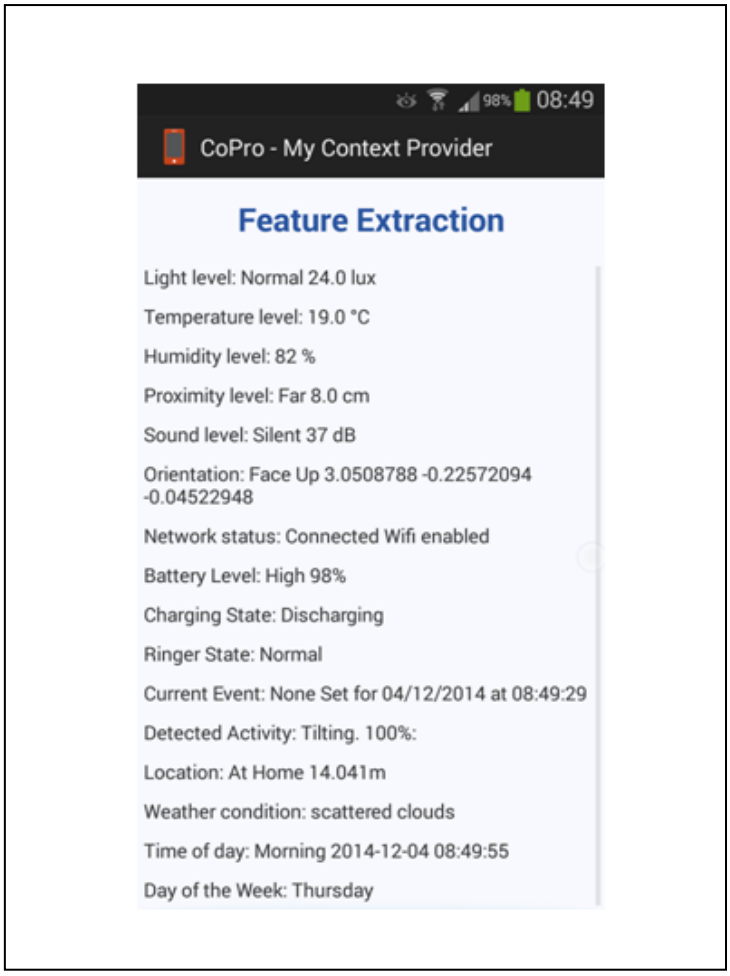

Figure 4.8: High-level contexts reported by CoPro using feature extraction

Two approaches were used to overcome this problem. The first mapped the preferences data to the extracted features. For example, the address feature for the current location was mapped on to the preference for the user's home. So if the predicted address matched the address specified in the preferences, this would then further infer that the user was at home. Making use of user preferences allowed the user to have some level of control over the context recognition and prediction process. This addressed the issue of balancing the user's control, identified in Section 2.5, with preferences while still performing automated context recognition.

The second approach was to combine the extracted features (i.e. high-level context) data with context rules to generate context models, as highlighted in Section 2.6, which inferred the specified context situations. Identifying these context situations of interest was highlighted as future work in Section 2.5. The context situation predication techniques and algorithms used

in CoPro for deriving inferred context are discussed in the next section. The inferred context situations that were determined are also identified in the next section.

4.3.4 Context Situation Prediction

The complex problem of predicting context situations identified in Section 2.5 was initially addressed using the first approach by means of preferences. The importance of preferences was emphasised during the analysis of existing mobile context models and infrastructures in Section 3.5. This analysis concluded (Section 3.7) that having user preferences was essential, with some context existing models, such as the one by Fausto and Alberto (2010) having two out of four core dimensions dedicated purely for user preferences. Supplementing context information with preferences was also highlighted in Section 2.2, which would help define the user, their behaviour and environment.

In order to obtain user preferences, Section 2.2 identified that user preferences could be explicitly provided from the user via the application user interface. For the purpose of this research a dummy Facebook account was created for testing purposes to simulate a user with a Facebook account. The use of the Facebook API identified in Section 4.3.1, allowed CoPro to obtain several preset preferences for the user by simply letting the user log in to Facebook via CoPro. This approach would require less effort from the user in providing their preferences, as opposed to the user entering each of their preferences manually.

Other preferences that could not be obtained from Facebook or if the user had no Facebook account would then have to be obtained directly from the user. In order to simulate this scenario, CoPro manually set two sets of preferences that would require input from the user. The first set of preferences were location preferences, including a work address and a home address. These location preferences were derived from Figure 3.8, which had a location context with values of home and workplace. Once these location preferences were set in CoPro, they were matched against the location context if a location address was obtained to determine if the location context was "At Home" or "At Work". The second set of preferences were derived from a medical PHR identified in Section 3.3.1. These medical preferences were combined with the Facebook preferences to form a single user profile of preferences. An example screenshot of the CoPro app is shown in Figure 4.9, which illustrates several of the preferences that were collected.

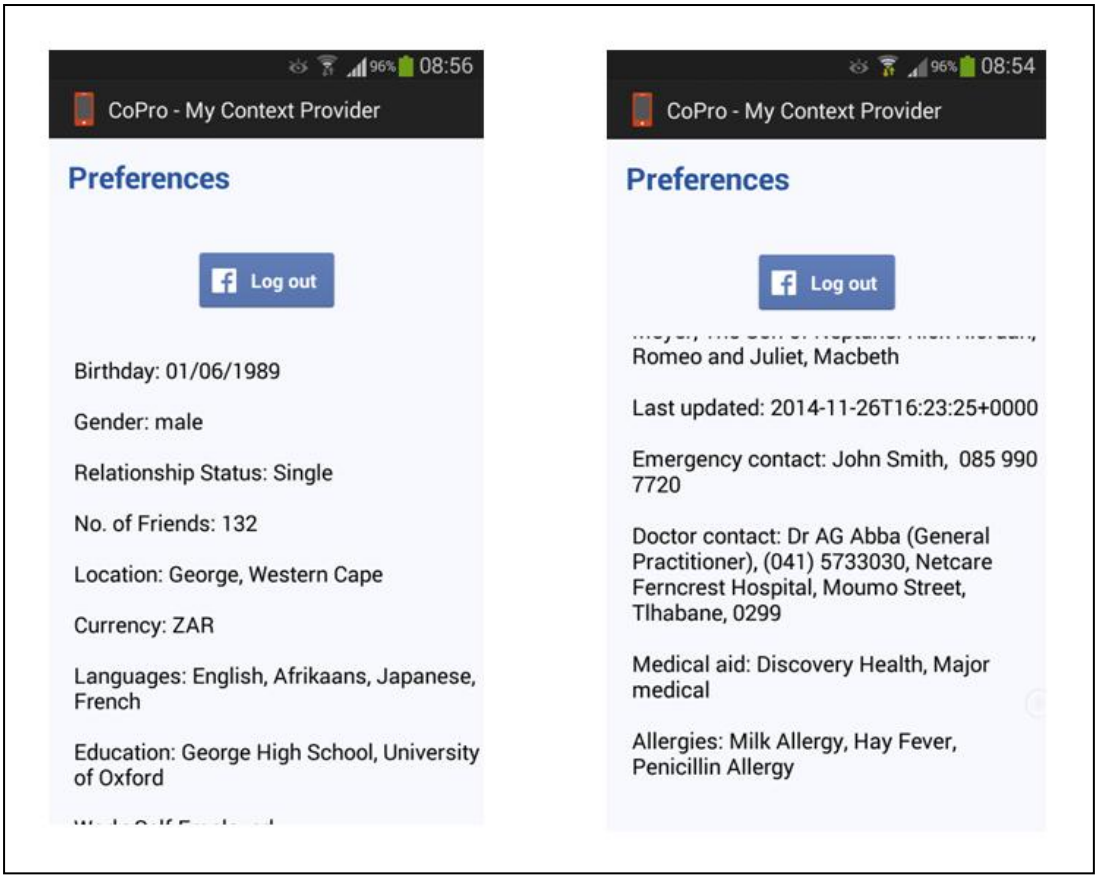

Figure 4.9: Example of user preferences obtained by CoPro

The following preferences were collected by CoPro:

- Preferences - Name, Birthday, Gender, Relationship status, Location, Currency, Languages, Education, Work, Movies, Music, Books.
- Medical profile - Emergency Contact (Name, number), Doctor (Name, number, hospital address), Medical Insurance (Medical aid, medical plan), Allergies.

The second approach used to deal with predicting context situations was to make use of inferred context rules. A total of five context situations of interest were identified and derived based on existing inferred context shown in Table 4.2 and the high-level context information highlighted in Section 4.3.3.

Table 4.2: Existing inferred contexts identified from literature

Paper Title	Identified Inferred Contexts
Context and Bio-Aware Mobile Applications	Place, Activity, Mood, OnPerson.
Extracting High-Level User Context from Low-Level Mobile Sensors Data	Home, School, Work.
Bayesian Network-Based High-Level Context Recognition for Mobile Context Sharing in Cyber-Physical System	Activity (Move, study, meal, sleep, sport, play, rest), Emotion, and User Relationship. "Related Work" Walking, Running, Idle and Resting.
An Ecosystem For Learning and Using Sensor-Driven IM Status Messages	Place (At lunch, out of the office, at home, library, in lab).
On Uncertainty in Context-Aware Computing: Appealing to High-Level and Same-Level Context for Low-Level Context Verification	In a Meeting, Sleeping.
Context Inference for Mobile Applications in the UPCASE Project?	Sleeping, Resting, Working, Meeting, Walking Outside, Walking Inside, Running Inside, Running Outside. Driving, Exercising.
Monitoring Natural Motions to Determine User Contexts and Intents	Walking Upstairs, Standing or Sitting.
CASS - Middleware for Mobile Context-Aware Applications	Rain Brightness Temp Goal Wet dull cold indoor
Activity Zones for Context-Aware Computing	Meeting, Reading.

Each paper and the associated inferred contexts are highlighted in Table 4.2. The existing inferred contexts are associated with either an activity, a location, a posture or a mood. As the high-level contexts of CoPro did not cater for mood, the mood inferred contexts were not taken into consideration.

The use of the produced high-level context to identify the five inferred context situations to be determined by CoPro, formed part of the context reasoning process highlighted in Section 2.4 in Chapter 2. The five inferred context situations that were identified are illustrated in Table 4.3.

Table 4.3: Identified inferred context situations

Inferred Context	Input sources	Values
1. Availability	Calendar, Activity, Ringer State, Network State.	Free, Semi-Busy, Busy, Unknown.
2. Indoor/Outdoor Location	Proximity, Location, Light, Calendar, Time of Day, Activity.	Indoor[H], Outdoor[H], Outdoor/Semi-outdoor[H], Indoor[L], Outdoor[L], Outdoor/Semi-outdoor[L]
3. Posture	Orientation, Proximity, Activity.	Upright, Lying Down, Sitting/Standing, Unknown.
4. Inferred Activity	Activity, Calendar, Location, Time of Day, Light, Sound, Proximity.	Sleeping, Resting, Working, In Meeting, Watching TV, Calendar Event, Going to Work, Going Home.
5. Device Location	Proximity, Activity, Charging State, Orientation.	On Charge, In Hand, In Pocket/In Bag, Unknown.

The inferred context situations shown in Table 4.3, are each paired with the input sources required for that situation's context rule to determine and report a value. The possible values that the context rule for each inferred context could produce are also listed in Table 4.3. Only the context rules for the second and fourth inferred context situations were developed based on existing context rules. The context rules for the remaining three inferred context situations were designed and implemented from this research. An example of the inferred contexts derived by CoPro is shown in Figure 4.10.

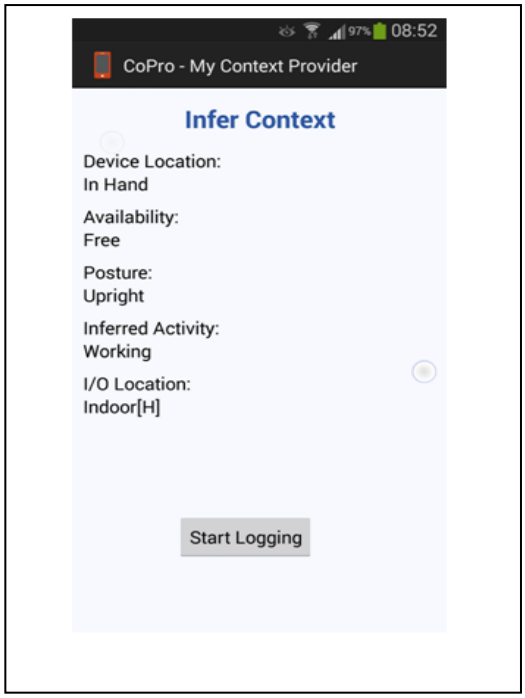

Figure 4.10: Example of user preferences obtained by CoPro

The first inferred context was availability, which was initially classified as a low-level context. However, as highlighted in Section 4.3.1, availability was a combination of two input sources - the activity and calendar. Therefore, availability was changed into an inferred context situation. Other factors were also later identified that could assist in determining the availability of the user, including the network and ringer state.

Indoor/Outdoor (I/O) location was identified as an inferred context situation as the lack of indoor positioning was one of the existing issues highlighted in Section 2.5 in Chapter 2. The I/O location context rule was designed by using an existing context rule as a basis (Zhou, Zheng, Li, Li & Shen 2012). The existing context rule used is shown in Figure 4.11.

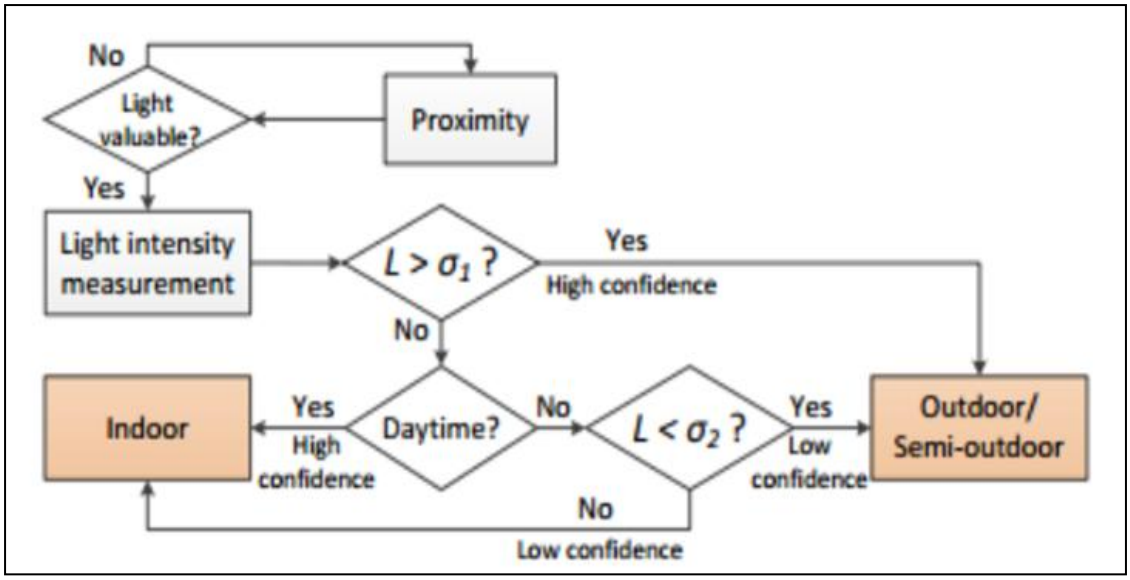

Figure 4.11: Existing context rule of light detector (Zhou *et al*. 2012)

The existing context rule was selected as a basis for the design of the context rule for the I/O location context as it used input sources that CoPro already supported. These input sources included the proximity, light and time of day. This context rule, however, only used three inputs and was therefore enhanced with three other inputs including the location, activity and calendar.

Posture was the third inferred context identified from the existing inferred contexts highlighted in Table 4.2. Posture refers to the posture of the user and was inferred by using the high-level values of orientation, proximity, activity.

Inferred activity was also identified from the existing inferred contexts highlighted in Table 4.2. Inferred activity represented a more abstract meaning (e.g. working) than a high-level physical activity (e.g. walking) identified in Section 4.3.3. Inferred activity's context rule used the most inputs in order to determine the actual inferred activity. These inputs included activity, calendar, location, time of day, light, sound, and proximity.

The final inferred context was the device location, which represents the location of the mobile device in relation to the user. The device location context was identified based on the high-level inputs and on the first row of existing inferred contexts highlighted in Table 4.2. The device location had the same inputs that were used for the posture context excluding the charging state.

4.4 Conclusion

This chapter addressed Research Question 3 that was identified in Chapter 1: "*How can an improved context-aware model be developed?"* This chapter discussed the design (Section 4.2) and implementation (Section 4.3) of the context-aware model, which was centred on the second cycle of the DSR methodology. The typical context awareness architecture (Fig 2.1) identified in Chapter 2 was extended (Section 4.2.3) to support context awareness in mobile applications using multiple input sources. The proposed context-aware model (Section 4.2.2) was based on the most complete existing context model that was selected in Chapter 3. The proposed model was then extended to address the shortcomings of the selected model. These extensions included the addition of context attributes identified in Section 3.6 that were not part of the existing context model and the addition of a health context. The contents of this chapter were incorporated in a paper presented at the Southern Africa Telecommunications Networks and Applications Conference in 2014 (Pather *et al*. 2014), supporting the feasibility (i.e. viability) of the artefact, which is an important aspect of DSR (Hevner *et al*. 2004).

The main deliverables of this cycle were the design of the context-aware model and the implementation of a prototype, called CoPro. The prototype was developed in an iterative manner, through which changes were made based on constant feedback as the prototype was implemented. CoPro was implemented effectively as it produced low-level, high-level and inferred context information simultaneously.

The next chapter will discuss the evaluation of the prototype to determine the extent to which CoPro can support the context awareness needs in mobile applications. The purpose of evaluating CoPro will be to assess the feasibility of the proposed model. CoPro will be evaluated in terms of utility, quality and efficacy by conducting several experiments. The results of these experiments will be presented, analysed and the findings will be used to refine the design and implementation of CoPro. The results will then be matched to the requirements from both literature Chapters 2 and 3 to determine the extent to which CoPro supported those design implications. Finally, design recommendations will be made based on the results and matched requirements to conclude the chapter.

Chapter 5: Evaluation

5.1 Introduction

The design and implementation of the proposed context-aware model as a prototype, called CoPro was discussed in Chapter 4. The design of the proposed model formed the basis for the implementation of the CoPro prototype. CoPro was implemented for the purpose of evaluating the feasibility of the proposed model.

This chapter addresses the evaluation phase of the DSR methodology and the fourth research question identified in Chapter 1: "*How effective, reliable and capable is the proposed context-aware model and to what extent does it support context awareness in mobile applications?"* As with the design and implementation of the proposed model, the evaluation of the proposed model was also a core activity of the Design Cycle in DSR. The Design Cycle allowed for re-evaluations to be conducted to refine the design artefact until the final design artefact was produced. Experimental evaluation methods, including controlled experiments and simulation, were used to demonstrate well-executed evaluation methods, as proposed by Hevner *et al*. (2004). The experimental evaluation methods rigorously demonstrated the utility, quality and efficacy of CoPro.

The chapter first identifies evaluation techniques, which can be used to evaluate CoPro. The goals of the evaluation are then described. A suitable evaluation technique is then selected and motivated based on the evaluation goals. The experimental design is then described in detail, which includes the evaluation objectives, evaluation metrics, evaluation instruments and evaluation procedure. The analysis of results as well as the design improvements for each evaluation are presented. Lastly, the design implications and the design recommendations are discussed to conclude the chapter.

5.2 Evaluation Techniques

Selecting the right evaluation techniques to evaluate the CoPro artefact was crucial in answering the fifth research question identified in Chapter 1. As CoPro is a DSR artefact, previous DSR literature could provide insight into which techniques would be the most appropriate for evaluating CoPro. A number of techniques for evaluating DSR artefacts have been identified by DSR researchers (Ostrowski & Helfert 2012).

March and Smith (1995) highlighted that evaluation should be a build and evaluate process. This process would involve the development of criteria and evaluating the artefact's performance based on the developed criteria. Determining not only if the artefact works but why and how it works was also emphasized by March and Smith (1995).

Hevner *et al*. (2004) proposed five evaluation techniques:

- Observational
    - Field Study - Observe use of artefact in multiple projects
    - Case Study - Examine artefact in depth in environment
- Analytical
    - Static analysis - Identify static qualities (e.g., complexity) by studying artefact structure
    - Dynamic analysis - Identify dynamic qualities (e.g., performance) by examining artefact while in use
    - Architecture analysis - Examine fit of artefact into technical architecture
    - Optimization - Show artefact's optimal properties or supply optimality bounds on artefact behaviour
- Experimental
    - Simulation - Execute artefact with artificial data
    - Controlled Experiments - Examine artefact in controlled environment for qualities
- Testing
    - Structural (White Box) Testing - Execute coverage testing of specific metric in artefact implementation
    - Functional (Black Box) Testing - Identify failures and flaws by testing artefact interfaces
- Descriptive
    - Scenarios - Demonstrate artefact's utility through well-designed scenarios
    - Informed Argument - Construct a convincing argument showing artefact's utility by using information from knowledge base (e.g., relevant research).

Peffers, Tuunanen, Rothenberger and Chatterjee (2007) suggested that evaluation should consist of two activities including demonstration and evaluation. The demonstration activity would involve showing that the artefact achieved its intended purpose and works in at least one context. The evaluation activity involves demonstrating how well the artefact solves the identified problem.

Venable (2006) also separated evaluation into two distinct and independent evaluation techniques including naturalistic and artificial. Testing the performance of an artefact in its

real environment is considered a naturalistic evaluation technique. A naturalistic evaluation includes several methods for evaluation such as action research, surveys, field studies and case studies. Artificial evaluation involves evaluating an artefact in a rigorously controlled, "scientific" manner. The artificial evaluation methods included in this technique are simulations, mathematical proofs, theoretical arguments, criteria-based analysis, field experiments as well as laboratory experiments.

5.3 Evaluation Goals

A key goal of DSR is to produce a viable artefact in the form of a construct, a model, a method, or an instantiation (Hevner *et al*. 2004). CoPro was developed with the goal of evaluating the proposed context-aware model. Determining the feasibility of this model by evaluating CoPro is aligned with the above mentioned DSR goal and is thus the main goal of the evaluation.

The proposed model focuses on the process of dealing with context awareness from multiple-input sources. This process forms the underlying middleware layer and context gathering and pre-processing layer, as illustrated in Section 4.2.3 of Chapter 4. These two layers produce and would ultimately provide context information to mobile applications to utilise.

Sensor-based context recognition was identified as an existing issue in context awareness in Section 2.5 of Chapter 2. Evaluating whether or not the CoPro prototype can actually perform sensor-based context recognition will validate if the proposed model works, as well as rigorously demonstrating the utility of the artefact (Ostrowski & Helfert 2012). Evaluating the effectiveness of CoPro in producing context information formed part of the main goal of the evaluation.

In order for mobile app developers to use the produced context information in their mobile applications, the context information needs to be reliable and useful. However, context information might not always be reliable or useful, which is seen as a problem related to the quality of the context information. The nature of context allows context information to be constantly changing, incomplete, uncertain, and obtained from multiple distributed input sources. To produce reliable context information when dealing with context awareness, the quality of the context information needs to be ensured (Kim & Lee 2006). The extent to

which the proposed model objectively provides quality context information also formed part of the main goal of the evaluation.

As the proposed model was aimed at mobile applications running on mobile devices, the limitations of mobile devices needed to be considered. These limitations included limited battery power, limited processing power and dealing with the absence of information, which was highlighted in Section 2.5. Evaluating how CoPro took these aspects into account will demonstrate how capable the proposed model is in solving the identified problem in Section 1.6. Determining the efficacy of the proposed model was the final part of the main goal of the evaluation.

The main evaluation goal together with each of the three sub-goals addresses the fifth research question identified in Chapter 1. The evaluation goals are also aligned with the DSR evaluation activity of the design cycle, in which the utility, quality, and efficacy of a design artefact must be rigorously demonstrated (Hevner *et al*. 2004).

5.4 Selection of Techniques

A strong basis for the selection process of the most suitable evaluation techniques was provided by matching the evaluation goals with the fifth research question identified in Chapter 1 and aligning those goals with the DSR methodology. Several evaluation techniques were identified in Section 5.2 from DSR literature; however, the DSR literature provides very little guidance in selecting amongst evaluation techniques and methods for evaluation in DSR (Pries-Heje, Baskerville & Venable 2008; Ostrowski & Helfert 2012).

Pries-Heje *et al*. (2008) proposed a framework to assist Design Science researchers in building strategies for evaluating their research outcomes as well as to achieve improved rigor in DSR. Their framework recommended looking into the two perspectives of general IS evaluation to provide some guidance in forming a strategy for DSR evaluation. The first perspective is *ex ante*, whereby potential technologies or systems are evaluated prior to being implemented, acquired or even selected. The second perspective is *ex post*, which involves evaluating the selected technology or system after it has been implemented or acquired. The *ex post* perspective seemed the most relevant as CoPro had already been implemented, based

on the proposed model and was ready to be evaluated. Yang and Padmanabhan (2005) classified *ex post* into four categories, as illustrated in Table 5.1.

Table 5.1: Categories of ex post evaluation methods (Yang & Padmanabhan 2005; Pries-Heje *et al.* 2008)

Computation of Quality Measures	**Setting**	
	Real	**Abstract**
Automatic	1. Experimental designs	3. Historic data experiments
Human Subjects	2. User opinion studies	4. Opinions analysis of historical data

As the proposed model was designed to support the context awareness process in mobile applications, it was appropriate to determine if the model could achieve this goal. Using the subjective opinions of users would not have validated if the proposed model worked from an objective point of view and thus eliminated the use of users for the evaluation. CoPro was developed with the intention of recognizing the current context. This intention suggested that CoPro needed to be tested within a real setting, which eliminated the abstract setting.

The first category of experimental designs was the only remaining category and appeared to match the designed and implemented artefact. Appropriately matching the designed artefact with the selected evaluation techniques and methods was the only guideline provided by Hevner *et al*. (2004). Experimental designs illustrates the collection and the calculation of quantitative data in real situations (Pries-Heje *et al*. 2008).

The experimental designs classified by Yang and Padmanabhan (2005) indicated that data collecting experiments would need to be conducted; however, they did not mention how this could be achieved. Matching experimental design with Hevner *et al*.'s (2004) experimental evaluation methods, namely controlled experiments and simulation, could assist in suggesting how these experiments should be done. Evaluating CoPro in a controlled environment would facilitate an objective evaluation. Simulation, as defined by Hevner *et al*. (2004), did not appear to be relevant at this stage as it is concerned with artificial data.

Peffers *et al*. (2007) also provided further insight that could be incorporated into the strategy for evaluating CoPro. Evaluating CoPro in at least one context would show that the proposed

artefact solves the identified problem in Section 1.6 of Chapter 1. The second activity of the two step process would involve demonstrating how well CoPro solves the identified problem.

Part of the guidelines suggested by Hevner *et al*. (2004) emphasized that the selection of evaluation techniques must not only be matched to the artefact but also to the selected evaluation metrics as well. The evaluation metrics that were selected stem from the Quality of Context Metrics (QoC) in Section 4.3.2 in Chapter 4.The selected evaluation metrics of the evaluation strategy for CoPro will be discussed in detail in Section 5.5.2. The evaluation metrics formed part of the design of the proposed model and were incorporated into the implementation of CoPro. Using and testing of the quality of context within the implemented model aligns with the structural testing evaluation method identified by Hevner *et al*. (2004). This testing method formed the final part of the strategy for evaluating the implemented artefact.

5.5 Evaluation Design

An experimental evaluation technique was selected as the most suitable technique based on the proposed artefact and was used for the evaluation of the artefact. Conducting experiments in DSR also supported the goal of objectively evaluating the overall quality of the artefact (Mettler, Eurich & Winter 2014). Ostrowski and Helfert (2012), however, emphasized the fact that DSR researchers are provided with very few examples of how one can actually execute evaluation at an operational level. Using the strategy formulated in Section 5.4 assisted in designing and conducting the evaluation at an operational level.

The evaluation design of the experimental evaluation technique was executed by performing controlled experiments in at least one context, which demonstrated if the artefact worked and how well it worked. The experiments collected, calculated and logged quantitative data for each context. The quantitative data was collected from inferred, high-level and low-level context values as well as from the structural testing (i.e. evaluation metrics) that was implemented in the artefact. The collected data was then analysed and the results were reported per evaluation conducted.

This section discusses the experimental evaluation that was conducted in two different contexts. Two different contexts were chosen to rigorously demonstrate the goals of the

evaluation that were highlighted in Section 5.3. As location awareness is an important aspect of context awareness as identified in Section 2.1.2, an indoor context and an outdoor context were selected. The selected contexts were related to the mobile user's physical location.

A significant aspect of using experiments in DSR evaluation is to evaluate the artefact under different conditions (Mettler *et al*. 2014). To ensure that the artefact was evaluated under different conditions, the device's location was selected as an independent variable based on the available inferred contexts. This independent variable was actively changed between being either in the hand or in the pocket of the evaluation administrator (the author). To ensure control during the experiments, the rest of the inferred, high-level and low-level context values were not actively manipulated. Mettler *et al*. (2014) support this by stating that only the independent variables are subject to deliberate manipulation in a controlled environment.

Combining the different independent variable states with the different contexts produced four distinct evaluation scenarios. Each scenario represented a real situation in which the proposed artefact can be used.

The following evaluation scenarios were used:

- Indoor - In Hand (IIH)
- Indoor - In Pocket (IIP)
- Outdoor - In Hand (OIH)
- Outdoor - In Pocket (OIP)

The state changes for all context values were collected and logged for each scenario. The evaluation metrics for each value were also collected, calculated and logged for each scenario. The data logged for both context values and evaluation metrics were then analysed with the results used to refine and improve the artefact. The artefact was re-evaluated and improved upon multiple times. This iterative evaluation process is a core characteristic of DSR, which improved the quality of the design and was iterated several times before the final design artefact was produced (Markus, Majchrzak & Gasser 2002; Hevner 2007).

5.5.1 Evaluation Objectives

The objective of the experimental design was to determine if the proposed model was feasible by objectively assessing the implemented artefact's ability to produce low-level, high-level and inferred context. Part of assessing CoPro's context recognition ability included whether it could produce context, the quality of the produced context and how it dealt with the limitations of context on a mobile computing platform. This assessment helped to identify design issues as well as implementation issues concerning the proposed model and CoPro. These issues could be factors that affected the artefact's ability to solve the problem it was intended to address of supporting context awareness in mobile applications. Identifying these issues as well as providing recommendations to overcome these issues thus formed a critical part of the evaluation objectives.

5.5.2 Evaluation Metrics

Mettler *et al*. (2014) identified that because there are no guidelines available for design experiments, Design Science researchers need to develop quality criteria for their experiments. This quality criteria will help other researchers understand the validity and reliability of the design experiments.

The evaluation metrics selected for the evaluation strategy for CoPro were derived from the QoC metrics in Section 4.3.2. Only the objective QoC metrics were selected for evaluation purposes in order to match the goals set out in Section 5.3. By only considering the objective QoC metrics, four QoC metrics remained, namely freshness, reliability, granularity and confidence interval, as illustrated in Table 5.2.

Table 5.2: Remaining objective QoC metrics for low level context

Quality of Context Metric	Measures	Source
Freshness	Indicates validity of context in terms of timeliness	Time period, Age
Reliability	Indicates the extent to which context can be considered credible	Freshness, Uptodateness
Granularity	Indicates the precision of the context	Max_granlevel, Cur_granlevel
Confidence Interval	Indicates the confidence in the context produced	Context provider

The freshness, granularity and confidence QoC metrics remained the same for the evaluation. Reliability, however, used up-to-dateness as an input source, which is a subjective QoC metric. The algorithm for the reliability metric was therefore changed to provide a more objective view of reliability. Zheng *et al*. (2012) suggested that reliability may be affected by the freshness of the context as well as the trustworthiness of the environment. Taking this suggestion into consideration and the lack of overall QoC metrics, additional QoC metrics were implemented as shown in Table 5.3.

Table 5.3: Additional objective QoC metrics for low/high level context

Quality of Context Metric	Measures	Source
Accuracy	Indicates the extent to which data is correct and reliable	Error of sensor, context values, t-distribution table
Trustworthiness	Indicates the trustworthiness of the context provider	Completeness, Context object
Completeness	Indicates the extent to which the available context information are present	Available context providers, Weightings of providers

Accuracy was defined as the extent to which context information was correct and reliable. However, it was difficult to identify the true value for the evaluated context information. A statistical estimation method identified by Kim and Keumsuk Lee (2006) was used to determine the accuracy of the context information. This method involved estimating a confidence interval for the sensor values produced. If the sensor value was within the confidence interval then that sensor value can be regarded as accurate. The method suggested by Kim and Keumsuk Lee (2006) was used as it was compatible with the underlying implementation of CoPro. As discussed in Section 4.3.2 in Chapter 4, a buffer was already used to capture the sensor values, which were needed for the statistical estimation method.

The estimation method was used to determine the accuracy of sensors that provided continuous context information such as the light sensor. In order to determine the accuracy, a root mean square error (RMSE) was calculated, which was then used to calculate the upper and lower limit of the confidence interval. The RMSE was calculated using Equation 7 (Kim & Lee 2006).

$$RMSE(S_i) = \sqrt{\frac{1}{N} * \left[\sum_{i=1}^{N} \left(x_i - \bar{x} \right)^2 \right]} \tag{7}$$

N represented the total number of sensor values in the buffer. x_i the current sensor value and $\bar{x}$ the mean of the sensor values in the buffer.

The next step was to calculate the confidence interval of the true value of a sensor S_i with which the current sensor value was compared. This confidence interval was calculated using Equation 8 (Kim & Lee 2006).

$$TV(S_i) = \left(\bar{x} - t(v,\alpha)\frac{\sqrt{V}}{\sqrt{N}}, \bar{x} + t(v,\alpha)\frac{\sqrt{V}}{\sqrt{N}} \right) \tag{8}$$

V represented the calculated RMSE value $t(v,\alpha)$ indicated the t-distribution with v = N-1 representing the degrees of freedom. By using $\alpha = 0.05$, $t(v,\alpha)$ was then obtained from the t-distribution table (Appendix A). $\alpha = 0.05$ was used as it represented a confidence of 95% on the t-distribution table. This 95% value indicated that at 95% confidence, the true value was within the estimated interval (Walpole, Myers, Myers & Ye 2002).

Trustworthiness was defined as the trustworthiness of the context provider that produced the context information. A trustworthiness rating was assigned to each context object that indicated the reliability of the provider of that context object. To remain consistent with the other QoC metrics, the rating assigned ranged between 0 and 1. For example the network context object was assigned a trustworthiness rating of 0.9 as its context provider supplied the network objects value as soon as it was detected. The closer the rating was to 1, the higher level of trustworthiness that context provider had in producing good quality context information. The trustworthiness was also re-calculated for context information, such as the

network context that had a completeness metric. The mean of the trustworthiness and completeness was used to determine a new value for trustworthiness.

Completeness was defined as the extent to which the available context information were present for a particular context object (Manzoor *et al*. 2008). The completeness QoC metric was only used to evaluate those context objects that had more than one context attribute and was calculated using Equation 9 (Manzoor *et al*. 2010).

$$C(\mathcal{O}) = \frac{\sum_{j=0}^{m} w_j(\mathcal{O})}{\sum_{i=0}^{n} w_i(\mathcal{O})} \tag{9}$$

The total number of the attributes of the context object `O` that had been allocated a value was symbolized by `m`. The weight of the jth attribute of `O` that had been allocated a value was represented by `Wj(O)` (Manzoor *et al*. 2008). The numerator was calculated by summing up each jth attribute multiplied by their assigned weighting. Similarly, the total number of the attributes of context object `O` was symbolized by `n` and `Wi(O)` represented the weight of the ith attribute of `O` (Manzoor *et al*. 2008). The denominator was calculated by summing up each ith attribute multiplied by their assigned weighting. The completeness QoC metric was then calculated by dividing the numerator with the denominator.

If `n` = `m` then the completeness would be equal to 1, which indicated that all the attributes of context object O had been assigned a value. For example, the network context object had a total of two context attributes including mobile data and Wi-Fi. The completeness of the network object would therefore be:

C(O) = (`m` * mobile weight + `w` * wi-fi weight)/(1 * mobile weight + 1 * wi-fi weight);

The availability of mobile and Wi-Fi in the above equation was represented as `m` and `w` respectively. In the denominator `m` and `w` were indicated as 1 to denote all attributes. The weights assigned to mobile and Wi-Fi indicate each attribute's significance to the total attributes. Both weights were assigned a weighting of 0.5 for the network context object as both the mobile and Wi-Fi enables internet connectivity and were therefore considered equally important.

Based on the new QoC metrics and the suggestion by Zheng *et al*. (2012), the reliability algorithm was reworked and was calculated for each streaming context value, such as the light values, using Equation 10.

$$R(O) = (\text{Trustworthiness} + \text{Freshness} + \text{Accuracy})/3; \quad (10)$$

For other context objects that are not streaming context values such as the network connectivity values, reliability was calculated by using Equation 11.

$$R(O) = (\text{Trustworthiness} + \text{Freshness} + 1)/3; \quad (11)$$

As the network connectivity was detected as soon as there was any change in value, the confidence interval value was set to 1, which represented 100% confidence in the value reported.

The above QoC metrics were used to evaluate the low/high-level context values. The QoC metrics that were used to evaluate the inferred context are shown in Table 5.4.

Table 5.4: Objective QoC metrics for inferred context

Quality of Context Metric	Measures	Source
Freshness	Indicates validity of context in terms of timeliness	Time period, Age
Completeness	Indicates the extent to which the available context information are present	Available context values, Weightings of context values
Certainty	Indicates the confidence in the context information produced	Freshness, Completeness

Freshness of inferred context values were calculated in same manner in which the freshness for the low/high-level context values were calculated. This interoperability of the freshness QoC metric by being able to be applied to all contexts, further highlights that the change made to the time period in Section 4.3.2 was significant. This change involved using the actual time between context readings instead of using minDelay. This change was also supported by the fact that the inferred context was produced with context rules and not a sensor, which produced the minDelay value.

Completeness was calculated in the same manner as with the completeness used for the low/high context values. Each attribute of each inferred context was assigned a weighting value and was used to calculate the inferred context's completeness.

Certainty was defined as the confidence in the context information produced. Certainty was calculated using Equation 12 (Kim & Lee 2006).

$$C(CxtObj) = \begin{cases} CO(CxtObj) \times \dfrac{NumberofAnsweredRequest + 1}{NumberofRequest + 1} \\ : if\ F(CxtObj) \neq 0\ and\ CxtObj \neq null \\ CO(CxtObj) \times \dfrac{NumberofAnsweredRequest}{NumberofRequest + 1} : otherwise \end{cases} \quad (12)$$

Certainty was calculated using the number of inferred context requests and the number of those inferred context requests that were answered. Certainty also used the freshness metric as well as the completeness of the context object. If the inferred context produced was considered fresh (i.e. F(O) > 0) at the time that certainty was calculated, the request for inferred context was considered as answered. The ratio of number of answered requests or replies to number of requests was multiplied by the completeness of the context object to calculate the certainty of that inferred context object.

All the QoC metrics discussed in this section were used as the evaluation metrics for evaluating CoPro. These QoC metrics were used to help determine the feasibility of the proposed model. Mettler *et al*. (2014) suggested that derived QoC metrics may represent a valuable contribution to DSR.

5.5.3 Evaluation Instruments

The main instruments used during the evaluation of CoPro consisted of two devices. The first device was the initial Samsung S4 identified in Section 4.3, which was used during implementation of the proposed model. The Samsung S4 was the primary device for the evaluation and was used in all the experiments conducted. The second device was a Blackberry Curve 8520, which was used as a timing device. The Blackberry device was used to time the individual runs of each experiment in order to ensure more control over the experiments. Utilizing a timing device for the experiments ensured that the data collected were not influenced by external factors. One factor would be when the evaluator came to

check to see whether that particular experiment run had completed. Checking the device before the experiment had completed could affect the results by detecting the evaluator's presence. For example, the proximity sensor could indicate a value representing something is near the device as the evaluator approached the device. The timing device therefore guaranteed that the device was only disturbed after each particular run was complete ensuring consistent data throughout each experiment.

Systematic logging was the final instrument that was used during the evaluation. Systematic logging was used to log each context value and the structural testing (i.e. QoC metrics) associated with each context value. The data were logged into two comma-separated values (CSV) files. CSV files were used as they supported tabular storing of the data in plain-text form (Dunwiddie 2014). The first file logged only the actual context values, which included inferred, high-level and low level context values. The second file logged the actual context values as well as the associated QoC metrics. Logging the data into two separate files allowed for easier analysis of the data. For example, analysing and identifying any discrepancies in the context values was easier using the first file as it only contained context values. Analysing the QoC metrics associated with those values in the first file were easier using the second file, as this file was structured to highlight the QoC metrics.

5.5.4 Evaluation Procedure

The experimental evaluation took place at the author's home. CoPro was not dependent on a specific location as the prototype was run on two mobile devices and thus could be used anywhere. However, to provide more control over the experiments the same location was used throughout the evaluation of CoPro. The location allowed for the experiments to be conducted both indoors and outdoors. Each experiment involved testing CoPro in the four scenarios identified in Section 5.5. CoPro was tested for three consecutive runs in each scenario, which entailed conducting a total of twelve test runs per experiment. Each scenario was tested at least three times to ensure that the context data collected were consistent. Each test run logged context data as mentioned in Section 5.5.3 and ran for five minutes. The context data were logged with a sampling rate of one sample per second. This sampling rate resulted in a total of three hundred rows of context values being logged per run in the first file. The second file contained each value in the first file on a separate row with the associated QoC metrics next to each value. As there were twenty-one context values logged

every second a total of three-thousand six-hundred rows of values were logged in the second file per run.

A total of four experiments were conducted by the evaluation administrator (the author) for the evaluation of CoPro. Each experiment was conducted separately and followed an iterative evaluation process. This process involved re-evaluating CoPro by performing one experiment after the previous experiments results were analysed and used to refine CoPro.

The first experiment focused on evaluating only the low-level and high-level context values. This was done in order to verify that the context values produced were feasible before performing experiments with the inferred contexts and QoC metrics. The first three experiments followed the same procedure. This procedure first required one device to be in the evaluation scenario either indoors or outdoors and then placed either in hand or in pocket. Once the device was in position, the logging of the current context was initiated by pressing of a button. The starting of the timing device commenced thereafter. The context data was then automatically logged for a period of five minutes before stopping. At this point, the timing device would notify the evaluation administrator that the logging process had finished. Once the logging process finished, CoPro provided a dialogue, which allowed the administrator to email the logged context data for further analysis.

This procedure was followed for each of the twelve runs per experiment of the first, second and third experiments. The third and fourth experiments, however, focused on evaluating all of the context values including low-level, high-level, inferred and QoC metrics. All the context data and QoC metrics were finally collected and analysed to be discussed in the evaluation results.

5.6 Evaluation Results

The evaluation results obtained from the collected and analysed data are presented in this section. The evaluation results were initially going to be presented per context dimension; however, because there was an inter-dependency of context variables between different dimensions they were not. By analysing the inter-dependency between context variables more significant results could be obtained than simply analysing the context variables per dimension.

All of the data collected was logged in tabular form into CSV files (samples of data from the fourth experiment are shown in Appendices B and C) as described in Section 5.5.3 and analysed using Microsoft Excel Version 2007. A total of 165,600 rows of data were collected for all experiments conducted. As the tabulated data was considered a reasonably large dataset, tools and techniques were needed to assist the analysis of this data. Pivot tables and samples of the tabular data were used to explain and support the motivation for the results obtained. A pivot table is a data summarization tool, which was selected as it was useful in analysing tabulated data and also supports the creating of cross tabulations (Phillips 2014). Cross tabulation is a statistical process that creates a contingency table by summarizing categorical data, which is frequently used in scientific research (Verial 2014). Samples of the tabular data were also used as they provided direct insight into the context data to identify significant results.

The results for the evaluation metrics (i.e. QoC metrics) used for the evaluation are discussed and interpreted in the next section. This section discusses the performance, quality and capability results for the evaluation of CoPro.

5.6.1 Utility Results

The utility results are discussed in terms of the effectiveness of CoPro. The aim of these results are to determine the utility of CoPro by providing evidence that CoPro can indeed perform sensor-based context recognition.

5.6.1.1 Effectiveness

The results that indicated the effectiveness of CoPro in supporting context awareness were identified by analysing the low/high-level and inferred data produced from three of the four experiments that were conducted. This analysis was focused on the process of producing context values in terms of obtaining initial context, detecting changes in context and tailoring context with the use of preferences. CoPro was tested in four evaluation scenarios consisting of three test runs per scenario, as discussed in Section 5.5.4.

The first experiment was first conducted in the indoor - in hand scenario. All the low/high-level context data were collected and logged as well as one inferred context variable; the device location. The device location was also collected and logged as this was the

independent variable selected in Section 5.5, and should report a matching value to the manipulated state (i.e. in hand). The results for the three consecutive runs of the indoor - in hand scenario are presented below.

The first run of the first experiment in the scenario (i.e. indoor - in hand) indicated the state of the independent variable as "In Hand" as expected. The first run also showed that CoPro could detect the initial context for 15 out of 16 context values as soon as the data logging commenced. No initial value was reported for the weather context, which continued for the remainder of the test run as shown in Table 5.5.

Table 5.5: Values for activity, location and weather for first run of first experiment (IIH)

Log	Activity	Location	Weather
1	Standing still. 100%:	At Home 16.97m	null
2	Standing still. 100%:	At Home 16.97m	null
3..	Standing still. 100%:	At Home 16.97m	null
..298	Standing still. 100%:	At Home 16.97m	null
299	Standing still. 100%:	At Home 16.97m	null
300	Standing still. 100%:	At Home 16.97m	null

The lack of weather data was due to the fact that the weather was only updated when the location was updated. As denoted in Table 5.5 the location was reported once and did not update for the remainder of the test run. Location was only updated when the device was moving (i.e. not Still), which was not the case as activity was reported as still throughout the test run. The dependency of weather on location and location on activity was part of the design and implementation of the proposed model, which was discussed in Section 4.3.1.

The reported location values as seen in Table 5.5 also indicate that the location address was successfully retrieved from the location's latitude and longitude coordinates. More significantly, the reported location values demonstrate the successful use and matching of preferences with the location address and therefore reported a high-level but more meaningful value of "At Home".

The context values for proximity, orientation, network, battery level, charging state, ringer state, calendar event and day of the week reported the same initial values throughout the test run. The reporting of the same initial values was expected as these variables were not actively manipulated or triggered. The light, temperature, humidity and sound context values

fluctuated throughout the test run even though they were also not actively manipulated or triggered. These context variables reported fluctuating values due to the nature of their context providers (i.e. hardware sensors) of actively sensing changes in the environment. The light values did not change by much starting at 41.0 lux and ending at 42.0 lux with fluctuations ranging between 39.0 lux and 45.0 lux, with 42.0 lux being the most frequently reported value, as shown in Table 5.6.

Table 5.6: Pivot table showing occurrences of light values for first run of first experiment (IIH)

Light value	Occurrences
39.0 lux	22
40.0 lux	50
41.0 lux	6
42.0 lux	112
43.0 lux	82
44.0 lux	26
45.0 lux	2
Total number of values	**300**

The temperature values started at 24.46 °C and ended at 24.25 °C with minor fluctuations indicating a slight drop in temperature. The humidity values, however, decreased over time starting at 59.72% and ending at 57.54%, signifying a more noticeable drop in humidity. The sound values fluctuated the most, ranging between 34.4 dB and 50.4 dB throughout the five minute run. The significant changes in sound values could indicate noise or that sound values change frequently due to the sensitive nature of the device's microphone. Overall no real signs of drift were identified; however, additional runs were needed to verify that there were no elements of drift as well as to identify the true cause of the constant sound fluctuations.

The second run of the first experiment (indoor - in hand) showed similar results to the first run including the minor variations in light and temperature values. The second run also highlighted a noticeable drop in humidity (from 59.74% to 56.96%). A significant phenomenon that did not occur in the first run was that temperature, humidity and activity took 3 seconds longer than other context variables to start producing values.

The third run showed similar results to the first run but also had the same phenomenon as in the second run, however only temperature and humidity took 3 seconds longer to start producing values. The delay in activity values could be caused due to the delay in connecting

with the activity client API. The temperature and humidity values are reported from the same sensor providing an explanation for the same delay in reporting values. The actual delay of temperature and humidity values are possibly due to a slow sensor start-up.

The sound values in both the second and third run as in the first run, fluctuated just as rapidly. Analysing a sample of the data illustrated in Table 5.7, during a significant change in sound values, provided some clarification for the rapidly fluctuating values.

Table 5.7: Sample of sound values for second and third run of first experiment (IIH)

Second run		**Third run**	
Log	**Sound value**	**Log**	**Sound value**
143	Silent 38.1 dB	13	Silent 36.5 dB
144	Silent 39.5 dB	14	Silent 44.4 dB
145	Moderate 56.8 dB	15	Silent 42 dB
146	Silent 50.5 dB	16	Silent 38.8 dB
147	Silent 45 dB	17	Silent 37.8 dB
148	Silent 36.5 dB	18	Silent 40.3 dB
149	Silent 41.1 dB	19	Silent 42.6 dB

During the second run, a significant jump in sound value occurred from the 144th value (39.5 dB) to the 145th value (56.8 dB). The subsequent values including the 146th (50.5 dB), 147th (45 dB) and 148th (36.5 dB) value indicate a gradual decrease in sound level. This decrease in sound level values after a significant jump in sound illustrates that a moderate ambient sound was indeed detected, which slowly dissipated. Sample values from the third run show a similar pattern. An increase in sound level occurred between the 13th and 14th values (36.5 dB to 44.4 dB), which then dissipated over the 15th (42 dB), 16th (38.8 dB) and 17th (37.8 dB) values. These sound level results illustrate the sensitive nature of the device's microphone in detecting changes in ambient sound level and thus account for the high frequency of sound value fluctuations.

All of the data collected showed no signs of drift as continued fluctuations in values were identified. However, even though humidity did have minor fluctuations in values, humidity values showed possible elements of drift as values started high then gradually decreased over time.

CoPro was then tested in the second scenario (indoor - in pocket) of the first experiment. The state of the independent variable was not reported as "In Pocket" as expected during the first and second run. The device location was instead reported as "Unknown", which was due to the orientation value not being "Face down" but rather "Face up" whilst in the pocket of the evaluator. As identified in Chapter 4, the "In Pocket" state was triggered when the orientation of the device was considered "Face down" and proximity denotes a value of "Near". This unexpected result highlighted that when the device was on its side as it was during the first two runs of the experiment in the "Indoor - In pocket" scenario, device location was not reported as "In Pocket".

Similarly to the first run of the first scenario, all three runs of the second scenario also showed that CoPro could detect the initial context values for 15 out of 16 context variables. This instance; however, no initial value was reported for the activity context instead of the weather context. For the first run the no value was reported for activity until the 13th second of the test run as shown in Table 5.8.

Table 5.8: Sample of activity values for first run of first experiment (IIP)

First run : Indoor - In Pocket (IIP)	
Log	**Activity value**
1..	null
..13..	Unknown. 38%:
..30..	Standing still. 97%:
..34	Standing still. 100%:

Activity values for all three test runs took longer than in the first scenario to stabilize and produce a "standing still" state. This delay was possibly due to initial device movement of when the device was placed into the pocket. This device moment was also probably responsible for the weather value being set from the start as opposed to no value in the first scenario.

The overall trend for temperature and humidity values were the same for all three runs. This trend was that temperature values increased while humidity values decreased. The humidity however, decreased by a far greater range than when in the "In Hand" scenario. For example the drop in humidity for the last run of the "In Hand" scenario was 3.62%, whereas the drop in humidity for the first run of the "In Pocket" scenario was 8.75%.

For the third run the device was placed face down to try ensure more control over the independent variable. The activity value was affected slightly as the value changed to tilting then unknown and then back to a standing still state. Activity changes took time to process but when the most probable state was detected it only took 4 seconds to validate the state with an accuracy from 77% to 100% for standing still state. The orientation also indicated a value of "Face Down", however, the placing of the device face down as opposed to on its side whilst in the pocket had no real effect on the data.

At the time when CoPro was to be tested in the third scenario (outdoor - in hand), the outside weather conditions were not ideal as it was raining. The last run of the previous test in the second scenario did suggest possible shower rain as a value of "light intensity shower rain" was reported for the weather context. Although the weather conditions were poor for testing CoPro on the mobile device, the test was still conducted with the mobile device covered for protection purposes. This covering of the device; however, did trigger the proximity sensor, which indicated a value of "Near" throughout the three runs in the outdoor in hand scenario. The shower rain weather condition was later confirmed by CoPro when the weather context reported a value of "shower rain" in the second and third run of the third scenario. To ensure more control over the experiments, the device was also placed on a table for subsequent test runs and experiments. With the device being face up when considered to be in hand and the device face down in the pocket of a pair of shorts when considered to be in pocket. This can be noted as a form of simulation, which was one of the methods for an experimental evaluation identified by Hevner *et al*. (2004).

A consistent result that was highlighted in all three runs, was that the temperature was considerably colder and the humidity was considerably higher outdoors as compared to indoors, as seen in Table 5.9.

Table 5.9: Sample of temperature and humidity values of first experiment (IIP and OIH)

	Indoor (IP)		Outdoor (IH)	
Log	**Temperature**	**Humidity**	**Temperature**	**Humidity**
1	Hot 27.18°C	Medium 59.28%	Mild 18.19°C	Medium 69.92%
2..	Hot 27.18°C	Medium 59.28%	Mild 18.19°C	Medium 69.92%
..150..	Very hot 30.27°C	Medium 48.84%	Mild 15.39°C	High 71.30%
..299	Very hot 30.79°C	Medium 46.60%	Cold 14.86°C	High 70.09%
300	Very hot 30.79 °C	Medium 46.60%	Cold 14.94°C	Medium 69.87%

The temperature and humidity values of the last indoor test run were compared with the first outdoor test run in order to compare the closest values in terms of logged time for both contexts. The initial temperature from the indoor run was classified as Hot 27.18°C, which increased and ended at Very Hot 30.79 °C. The temperature then dropped by 12.6°C for the outdoor run, with an initial value classified as Mild 18.19°C, which decreased and ended at Cold 14.94°C. The humidity values on the other hand increased when comparing the indoor run to the outdoor run. The humidity increased by 23.32% when comparing the last reported humidity value of the indoor run and the initial humidity value of the outdoor run. Overall these results showed that the temperature was colder and humidity was higher outdoors as compared to indoors.

The only other significant changes for the third scenario (OIH) involved the sound context. The sound values reported were moderately higher when outdoors with 169 occurrences of a moderate fuzzy value than when indoors, which reported only 2 occurrences of the moderate fuzzy value. A number of factors that could have affected the results for this scenario include the weather condition, which was reported as "shower rain" and the device location, which was in the evaluator's pocket for the indoor scenario.

The fourth and final scenario involved testing CoPro outdoors with the device in pocket. As highlighted when discussing the third scenario, the device for this scenario was placed on a table face down in the pocket of a pair of shorts to simulate the device being in pocket. The most noticeable change as compared to previous scenarios, was delays in reporting initial

context values for temperature, humidity, activity of about 2 to 3 seconds and weather of about 3 to 5 seconds.

In order to verify the results in the first experiment a second experiment was conducted with the more controlled and simulated approach. All the low/high-level context data were collected and logged as well as one inferred context variable, the device location. The results for the second experiment were compared to the results of the first experiment to identify any noteworthy discrepancies and changes.

The second experiment was first conducted in the outdoor in-hand scenario. Device location was reported as "In-Hand" as expected as compared to "Unknown" in the first experiment. Weather reported a value of "Sky is clear" as opposed to "shower rain". The temperature values had minor fluctuations instead of decreasing. Humidity values, however, decreased as opposed to increasing over time. The light values were reported as "Very bright" instead of "Very dark". These changes in reported values could be a result of the different weather conditions as well as the time of day as the first experiment was conducted at night and second experiment during the day.

Similarly, for the outdoor in-pocket scenario, humidity decreased over the 5 minutes (from 59.39% to 48.42%) with small fluctuations of ±1%. Temperature in this case increased (from 24.82°C to 30.22°C) with small fluctuations of ±1°C. The light values showed that the values decreased considerably by 45142 lux from being in-hand to in-pocket. A noteworthy observation, however, was that the light values still reported a value of "Bright" ranging between 170-156 lux for the first run. The second and third runs reported light values of "Normal" between 35-29 lux and 27-19 lux. This observation is significant as one would expect the light values to indicate a value of either "Dark" or "Very dark" when face down and in the pocket of a pair of shorts, which it did not. This significant observation was used to help improve the context rules for determining the outdoor and indoor position of the user by using the resultant light value ranges with the time of day.

For the indoor in-hand scenario no significant changes were discovered from both sets of results from the first two experiments. For example, the delays in reporting the initial temperature, humidity as well as activity values were consistent with the first experiment.

For the final scenario indoor in-pocket of the second experiment, the device location was reported as in-pocket as expected, which was a result of using a more controlled and simulated approach. The only other significant result was that between the 10th and 14th second of the first test run Wi-Fi/internet connectivity was lost and only restored at the 15th second of the run. This resulted in no weather value being reported until the 15th second of the run when Wi-Fi/internet connectivity was restored.

As the first and second experiment provided sufficient results to support the effectiveness of CoPro in terms of low/high-level context, further analysis focused on the inferred contexts. The results of these inferred contexts of the third experiment are highlighted below.

The third experiment was first performed in the outdoor - in hand scenario, followed by the outdoor - in pocket, the indoor - in hand and finally the indoor - in pocket scenario. All the low/high-level and inferred context data were collected and logged. The inferred results for the three consecutive runs of all four evaluation scenarios are presented below.

All three runs of the outdoor - in hand scenario consistently reported the same values for four of the five inferred context variables from start of the run to the end as shown in Table 5.10.

Table 5.10: Sample of inferred context values of first run of third experiment (OIH)

Log	Device Location	Availability	Posture	Indoor/Outdoor Location
1..	In Hand	Free	Upright	Outdoor/Semi-outdoor[H]
..150..	In Hand	Free	Upright	Outdoor/Semi-outdoor[H]
..300	In Hand	Free	Upright	Outdoor/Semi-outdoor[H]

All five of the inferred context values were determined by the context rules described in Section 4.3.4 in Chapter 4. The device location state was reported as "In Hand", which was expected as device location was the independent variable. The availability was described as "Free", which was correct and expected as there was no calendar event specified for that time period and the device remained still for all three test runs. An "Upright" value for posture was also an expected result as the orientation was described as "Face Up" and the proximity level was expressed as "Far". The indoor/outdoor location was described as "Outdoor/Semi-outdoor [H]", which indicates that the device was detected outdoors or semi-outdoors with a high confidence based on the context rule used for Indoor/Outdoor (I/O) location. A semi-outdoor value was paired with the outdoor value as a semi-outdoor context can represent

similar conditions to an outdoor context. For example, when a user is on a balcony the context conditions will denote the user location as being outdoors, which could be better represented and thus a semi-outdoor state was introduced for the indoor/outdoor location context as identified in Section 4.3.4.

Inferred activity, the fifth inferred context, was initially described as "Working" in all three test runs, however changed back and forth between "Resting" and "Working". These changes in inferred activity occurred when the sound values fluctuated between "Silent" and values classified louder than "Silent" as illustrated in Table 5.11.

Table 5.11: Sample of inferred activity values of second run of third experiment (OIH)

Log	Inferred Activity	Sound value
50	Working	Moderate 62.1 dB
51	Working	Moderate 65 dB
52	Resting	Silent 56 dB
53	Working	Moderate 58.1 dB
54	Resting	Silent 56 dB

The inferred activity state changes were correct based on the context rule used to determine this inferred context. However, these jumps between values indicate that it can be difficult to determine the inferred activity that matches the true context.

The outdoor in-pocket scenario highlighted three key results regarding the posture, inferred activity and I/O location. For all three runs the posture context was indicated as "Unknown", which was caused by the context rule for posture not handling the situation when proximity was expressed as "Near" and orientation was represented as "Face Down". This result was used to improve the design and implementation of the context rule used for inferring the posture context. Similarly to the first scenario, the inferred activity was affected by the fluctuating sound values but were also affected by the activity context when it was reported as "Tilting". This result showed that even though determining the inferred activity can be considered difficult, multiple inputs were used to try and address this problem. Finally the last key result regarding the I/O location demonstrated the improvement made to the I/O location context rule, which was identified in the second experiment (OIP). The improved I/O context rule uses a combination of the light (Bright and Normal), proximity (Near) and time of day (Morning and Afternoon) values when inferring I/O location.

For the indoor-in hand scenario inferred contexts reported expected values for the scenario; however, key relationships between inferred contexts and inputs for inferred contexts were identified. These relationships were identified when context values were missing, as shown in Table 5.12.

Table 5.12: Sample of context values of second run of third experiment (IIH)

Log	Inferred Activity	I/O Location	Light	Temperature	Humidity	Activity	Location
1	null	null	null	null	null	null	null
2	null	Indoor[H]	Bright 162.0 lux	null	null	null	At Home 23.0m
3	null	Indoor[H]	Bright 161.0 lux	null	null	null	At Home 23.0m

By analysing the data from the second run denoted in Table 5.12, a possible relationship can be seen between the inferred activity and temperature, humidity and activity values. However, by analysing the data from the third run shown in Table 5.13, it was clear that inferred activity only has a relationship with activity values. This was highlighted by the fact that a value for inferred activity was still reported even though both temperature and humidity did not report any values (i.e. null).

Table 5.13: Sample of context values of third run of third experiment (IIH)

Log	Inferred Activity	I/O Location	Light	Temperature	Humidity	Activity	Location
1	Resting	Indoor[H]	Bright 238.0 lux	null	null	Still. 100%	At Home 23.0m
2	Resting	Indoor[H]	Bright 238.0 lux	null	null	Still. 100%	At Home 23.0m
3	Resting	Indoor[H]	Bright 241.0 lux	null	null	Still. 100%	At Home 23.0m

Another possible relationship from Table 5.12 can be identified between the I/O location and light and location values. The data from the third run presented in Table 5.13, does not provide assistance in determining the true relationship as both light and location provide values. On the other hand, by analysing the context rule in Section 4.3.4 for the I/O location it was clear that the light context has a relationship with the inferred I/O location. The relationship between inferred activity and activity was also verified when analysing the context rule for the inferred activity.

The final scenario, the indoor in-pocket scenario, highlighted the same key result regarding the posture context, which was described as "Unknown" for all three runs. The other major result was that the I/O location context reported no value (i.e. null) during all three runs of the IIP scenario. After analysing the context rule that infers the I/O location context, it was identified that the context rule did not cater for the conditions detected in all three runs. As a result this finding was used to further refine and improve the context rule in inferring the I/O location.

The results presented indicated the effectiveness of CoPro in supporting context awareness, which was identified by analysing the low/high-level and inferred context data produced from the first three of the four experiments that were conducted. The analysis of the first three experiments, demonstrated that CoPro could not only produce context values in terms of obtaining initial context but also detect changes in context as well as tailor context using preferences.

5.6.2 Quality Results

The quality results described the reliability of the context data produced by CoPro. The evaluation metrics helped in determining the quality of the context and thus the extent to which the context produced was reliable and useful.

5.6.2.1 Reliability

The results that demonstrated the reliability of CoPro in supporting context awareness were identified by analysing the QoC metrics of the context produced from the third and fourth experiments that were performed. The focus of this analysis was on the quality of the context values produced in terms of freshness, reliability, granularity, confidence interval, accuracy, trustworthiness, completeness and certainty. CoPro was tested for two experiments in four evaluation scenarios consisting of three test runs per scenario, as discussed in Section 5.5.4.

The QoC metrics used to evaluate the low/high-level contexts included freshness, reliability, granularity, confidence interval, accuracy, trustworthiness and completeness. For assessing the quality of the inferred contexts the QoC metrics freshness, completeness and certainty were used.

The third and fourth experiment collected and logged both context values as well as the associated QoC metrics for each of the context values. The QoC metrics that were used for both the low/high-level and inferred contexts were analysed first, followed by the QoC metrics specific to the low/high-level contexts. Finally the QoC metrics specific to the inferred contexts were analysed last. The QoC values for all metrics for each run of every scenario were combined and summarized using a consolidated Excel PivotTable, which contained all the data (i.e. three runs of data) for that specific scenario. The consolidated PivotTable was then filtered, analysed per QoC metric and the results for the third and fourth experiments are presented in Tables 5.14 to 5.21 below.

For all the low/high-level and inferred contexts, freshness was the first QoC metric that was analysed. Freshness was calculated by using the age of the context and the time period as described in Section 4.3.2. Age of the context represented the difference between the current time and when that context was measured. Time period was identified from literature as the time interval between two readings (Manzoor *et al*. 2008). For example, if the first context reading was taken now and the second context reading was taken five seconds later, the time period would be five seconds. Although there are several time period definitions discovered in literature (Section 4.3.2), the definition of time period by Manzoor *et al*. (2008) was used as they distinguished between both the objective and subjective nature of context. The time period was therefore represented by the difference between the second context's measurement time and the first context's measurement time. The freshness results for all context values from both experiments are illustrated in Table 5.14.

Table 5.14: Pivot table of freshness results for all runs of third and fourth experiments

	Freshness			
	3rd Experiment		**4th Experiment**	
Scenario	**0**	**1**	**0**	**1**
OIH	11189	7083	6533	11757
OIP	11087	7787	6145	11823
IIH	11235	7038	6152	12112
IIP	11044	7521	6818	12010
Total	44555	29429	25648	47702

The freshness values from both experiments were only represented as either 0 or 1, with 0 indicating that the data was not objectively fresh and 1 demonstrating that the data was

indeed fresh. The results in Table 5.14 highlight the occurrences of each particular freshness value (i.e. 0 or 1).

The results for the third experiment show that freshness reported more values that were regarded as not fresh, with a total of 44555 occurrences of 0 compared to 29429 occurrences of 1. This result could be due to the actual data not being fresh or the algorithm used not correctly calculating the freshness. The latter was assumed and thus an assessment of the freshness algorithm was done. By assessing the algorithm used, it was discovered that the value calculated to represent the time period was not producing the correct value.

A significant problem that was identified from the lack of QoC literature regarding the implementation of time period, was how the most optimal value for time period could be obtained. For example, by using the sensor configuration (i.e. minDelay provided by sensor) or calculating the time period once and then using the same value repeatedly for all of those specific context readings or calculating the time period dynamically at run time using a unique time period per value pair (i.e. two values). (Manzoor *et al*. 2010; Zheng *et al*. 2012) suggest the first option and possibly the second for obtaining a value for the time period; however, none of them have provided evidence in testing the freshness formula with the time period that they defined. Therefore, none of the above literature sources have validated if the time period as defined by them worked when calculating the freshness of context. Another important issue that needed to be taken into consideration was the time period for the first context value obtained. The definition for time period highlighted the time between two values but did not consider the first value.

The first option of using the minDelay provided by the sensor would not be an ideal value for time period as this is the minimum time period and not the actual time period between reported context values (Google Inc 2014f). The first option is also limited as it can only be used for contexts provided by sensors. The second option is also flawed as it would involve consistently using the same time period for all value pairs, which in reality have varying time periods. The third option is the best suited for obtaining the actual time period for each individual value pair.

After an extensive process of reworking the way that the time period was determined, a possibly accurate algorithm for calculating the time period was produced. The final process used to identify how the time period was calculated is illustrated in Figure 5.1.

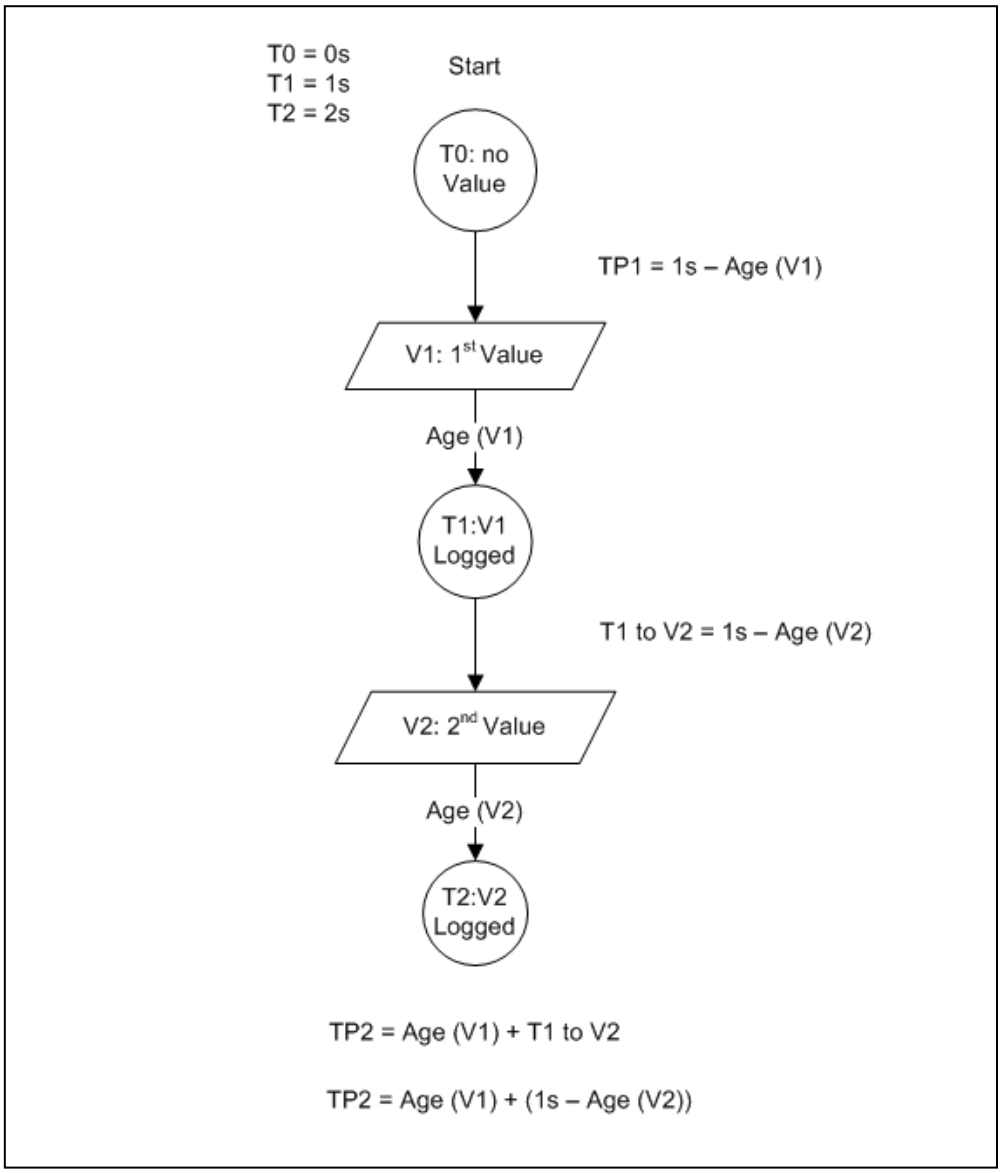

Figure 5.1: Final process used to calculate time period for context values

The final process in Figure 5.1 identified how the time period should be calculated, which was used in calculating the freshness of context. Not only was the algorithm for dynamically calculating the time period, identified but also how to calculate the time period for the first context value was identified.

The time period for the first value (TP1) was calculated not based on the time interval between two values but on the time interval from the start of the context recognition process

(T0) to the first value obtained (V1). Based on the logging sampling rate of 1 second as mentioned in Section 5.5.4, the age of the first value was the time interval between when the first value was obtained (V1) and when that first value was logged (T1). TP1 was then calculated by subtracting the age of the first value (Age (V1)) from 1 second, which was represented as 1000 milliseconds to match the unit of the age value. An example of the code that calculates the time period of the first I/O location value is shown in Figure 5.2 as Code Extract 1.

```
(1)     Line 1 - curr_ioloc.timePeriod = 1000 - curr_ioloc.age;

(2)     Line 1 - long prevAge = curr_ioloc.getAge(curr_ioloc.cMeasuredTime);
        Line 2 - curr_ioloc.updateTime(ct, this);
        Line 3 - curr_ioloc.setAge(curr_ioloc.cMeasuredTime);
        Line 4 - long timeP = prevAge + (1000 - curr_ioloc.age);
        Line 5 - curr_ioloc.timePeriod = timeP;
```

Figure 5.2: Code extract of code used to calculate time period for I/O location

The time period for the subsequent values (TP2) was dynamically calculated based on the time interval between value pairs (V1 and V2). From Figure 5.1 it is clear that the time period between V1 and V2 was equal to Age (V1) plus the time interval between T1 and V2. This time interval between T1 and V2 was calculated by subtracting the age of the second value (Age (V2)) from 1 second (i.e. 1000 milliseconds). The time period between V1 and V2 was therefore calculated by adding the age of the first value (Age(V1)) to the period between T1 and V2. An example of the code that dynamically calculates the time period for subsequent I/O location values is shown in Figure 5.2 as Code Extract 2.

The code extract (Figure 5.2) is an example of how the new time period was calculated, which in this example would be used for calculating the freshness of the Indoor/Outdoor location. Line 1 shows the code used to keep a reference to the age of the previous context value. The current I/O location object's measured time is then updated with the current time (i.e. ct) in Line 2. Line 3 sets the new age for the current I/O location object with the new

measured time. The time period was then calculated by adding the age of the previous context object to the difference between 1000 (1 sec in milliseconds) and the current age in Line 4. Finally the current I/O location object's time period was set in Line 5.

Another assessment of the freshness algorithm showed that it did not consider the situation when the age of the context and time period were equal. Therefore, the freshness algorithm was also improved to deal with the situation age = time period, whereby the freshness of the context would still be considered fresh and assigned a freshness value of 1. The values from the third experiment also revealed that the age of certain values remained the same even though they did not change. A change of re-calculating the age of context objects over time was implemented to ensure that the age of context objects increased if no change in value had occurred to represent the true age of those context objects.

The above changes were made to CoPro before the fourth and final evaluation experiment was conducted. The fourth experiment helped to determine whether these changes were actual improvements based on the results obtained from the final experiment.

The results for the fourth experiment as illustrated in Table 5.14 highlight that more values were considered as fresh at the time they were logged, with a total of 47702 occurrences of 1 compared to 25648 occurrences of 0. These results compared to the freshness results of the third experiment show an improvement of 25.26% in freshness. These result demonstrate that the changes made to the freshness and time period algorithm, and re-calculating of age are in fact significant improvements made to the design of CoPro.

Completeness was another shared QoC metric that was used for some low/high-level contexts and all inferred contexts. Completeness was calculated as described in Section 5.5.2, which uses the available context attributes and the weightings assigned to those attributes. The completeness results are presented in Table 5.15 for both the third and fourth experiments. These results were categorized into ranges to illustrate the results more clearly.

The completeness results from the third experiment highlighted that the overall completeness for most of the context values evaluated (71%) were reported as 1 with 17880 occurrences. Seventy one percent of the values reported a completeness value of 100%, which indicated that those values met the completeness requirements.

Table 5.15: Pivot table of completeness results for all runs of the third and fourth experiments

	Completeness							
	3rd Experiment				4th Experiment			
Scenario	**0.3**	**0.5**	**0.5 - 1**	**1**	**0.3**	**0.5**	**0.5 - 1**	**1**
OIH	900	900	65	4435	900	900	0	4500
OIP	900	900	15	4485	899	903	21	4173
IIH	900	905	20	4475	900	900	28	4463
IIP	900	900	15	4485	899	903	127	4361
Total	3600	3605	115	17880	3598	3606	176	17497

As the results for the third experiment did not highlight any problems no further improvements were made to the completeness algorithm. Comparing these completeness results with those of the fourth experiment, no significant changes were identified as the results appeared to be similar with some variation as presented in Table 5.15.

The reliability QoC metric was only used for all the low/high-level contexts. The new more objective reliability metric formula as described in Section 5.5.2 was used in both the third and fourth experiments. The evaluation results of reliability are presented in Table 5.16.

Table 5.16: Pivot table of reliability results for all runs of the third and fourth experiments

	Reliability					
	3rd Experiment			4th Experiment		
Scenario	**0 - 0.5**	**0.5 - 0.75**	**0.75 - 1**	**0 - 0.5**	**0.5 - 0.75**	**0.75 - 1**
OIH	4486	3565	5721	1625	5121	7044
OIP	4493	3636	6245	1615	4799	7357
IIH	4494	3660	5619	1614	4714	7445
IIP	4184	3780	6101	1610	5332	7395
Total	17657	14641	23686	6464	19966	29241

The reliability results of both experiments were categorized into three ranges including 0 (incl.) to 0.5, 0.5 (incl.) to 0.75 and 0.75 (incl.) to 1 (incl.). The third experiment showed that 31.54% of the results were in the first range, 26.15% of the results were in the second range and 42.31% of the results were in the third range. These results indicate that more than half of the results (57.69%) had a reliability of less than 75%. On the other hand the results from the fourth experiment showed that 11.61% of the results were in the first range, 35.86% of the results were in the second range and finally 52.53% of the results were in the third range.

Comparing the reliability results of both the third and the fourth experiments, highlighted a noticeable improvement in each range, with 19.93% less results in the first range (0-0.5), 9.71% more results in the second range (0.5-0.75) and 10.22% more results in the third range. The fourth experiment had 88.13% of its values above 50% reliability as opposed to 68.46% of the third experiment.

This improvement shows that values in the fourth experiment were considerably more reliable than the values of the third experiment. The reliability improvement from experiment three to four was due to the fact that the QoC metric reliability was dependent on freshness. This dependability is significant as the calculation of freshness as a whole was improved after the results of the third experiment was analysed. These enhanced reliability results in the fourth experiment provide further evidence that the previous changes made to CoPro improved the overall design and implementation of the proposed model.

Granularity was only evaluated for contexts with multiple levels of precision, which included the location context as well as the streaming sensors consisting of light, temperature, humidity and sound context. The granularity was calculated based on the precision of the actual data compared to the maximum precision that specific value could achieve. For example, a sound value of 32.22 dB would have a granularity of 1 if the maximum precision for sound was 3. The maximum precision value of sound, which is 3 in this example, is calculated by adding 1 to the max number of decimals a sound value could contain (i.e. 2).

Table 5.17: Pivot table of granularity results for the third and fourth experiments

	Granularity					
	3rd Experiment			**4th Experiment**		
Scenario	**0 - 0.5**	**0.5 - 0.9**	**0.9 - 1**	**0 - 0.5**	**0.5 - 0.9**	**0.9 - 1**
OIH	1800	553	2137	0	441	4047
OIP	1800	529	2163	0	885	3598
IIH	1800	570	2113	0	532	3951
IIP	1800	533	2155	0	531	3949
Total	7200	2185	8568	0	2389	15545

The granularity results of the both experiments as highlighted in Table 5.17, were broken up into three ranges including 0 (incl.) to 0.5, 0.5 (incl.) to 0.9 and 0.9 (incl.) to 1 (incl.). The third experiment had the following results per range: 40.11% (first), 12.17% (second) and 47.72 (third). After analysing the results of the third experiment it was clear that some values

incorrectly reported a granularity value of 0. The location and sound values were identified as reporting a granularity value of 0. This result was due to the granularity for each location and sound value not being set. Changes were made to ensure that the granularity for both the location as well as the sound contexts were set at run time. The changes made to CoPro after analysing the results of the third experiment were evident in the results of the fourth experiment. The fourth experiment provided evidence that the changes improved the quality for granularity values produced by having no values report a granularity value of 0. A more significant improvement was that 86.86% of the granularity results for the fourth experiment were between 90 and 100% in comparison with the granularity results of the third experiment, which were only 47.68%.

The confidence interval was only obtained for the location and activity context as the context providers of these input sources provided a confidence interval with the context data. Therefore no calculation of the confidence interval was needed. These results for the confidence intervals of the third and fourth experiments are presented in Table 5.18.

Table 5.18: Pivot table of confidence interval results of the third and fourth experiments

	Confidence Interval							
	3rd Experiment				**4th Experiment**			
	Activity		**Location**		**Activity**		**Location**	
Scenario	**0.3 - 0.9**	**0.9 - 1**	**< 20m**	**>= 20m**	**0.3 - 0.9**	**0.9 - 1**	**< 20m**	**>= 20m**
OIH	0	887	4	896	0	900	600	300
OIP	56	841	579	321	19	875	0	899
IIH	0	895	0	900	0	895	300	600
IIP	11	886	0	900	12	863	0	899
Total	67	3509	583	3017	31	3533	900	2698

The confidence interval results for the third and fourth experiments were tabulated into ranges of 0.3 (incl.) to 0.9, 0.9 (incl.) to 1 (incl.) for Activity and less than 20m, greater than or equal to 20m for Location. The third experiment's results showed that most activity values (98.14%) were reported with a confidence of 90 to 100%. The location results of the third experiment highlighted that 83.81% of the location values were reported with a confidence of greater than or equal to 20m. The remaining 16.19% of the location values reported a confidence of below 20m. Compared to the results of the fourth experiment there was a slight improvement in confidence for both activity values above 90% (99.13%) and location values below 20m (25.01%). No changes were made to CoPro after the third experiment that would

have affected the confidence interval results of the fourth experiment. Overall the activity values reported by CoPro were fairly accurate with 98.63% of them reported with a confidence above 90%. Although the overall location values reported only 41.2% of values with a confidence of below 20m, the majority of the values accurately reported a location address that matched the true value of location (i.e. At Home).

The accuracy QoC metric was used to evaluate the quality of the low/high-level contexts including light, temperature, humidity, sound and orientation. Accuracy was only used for these contexts as they were collected in a buffer, which contained an array of values for each context needed for the statistical method described in Section 5.5.2. The identified statistical method was used to calculate the accuracy QoC using the error of the sensor and an array of sensor values. The results for both the third and fourth experiments are presented in the consolidated Pivot table below (Table 5.19).

Table 5.19: Pivot table of accuracy results of the third and fourth experiments

	Accuracy			
	3rd Experiment		**4th Experiment**	
Scenario	**0**	**1**	**0**	**1**
OIH	206	4284	213	4275
OIP	203	4289	259	4225
IIH	372	4112	182	4301
IIP	218	4270	125	4356
Total	999	16955	779	17157

The results of the third experiment indicated that 94.44% of the context values were reported as accurate by having an accuracy value of 1, with only 5.56% reported as not accurate by having an accuracy value of 0. Similar results were obtained from the fourth experiment with 95.66% of values considered accurate and only 4.34% of values determined as not accurate. This minor improvement from the third to the four experiment is not significant as no changes were made after the third experiment that could have affected the accuracy results of the fourth experiment. Overall the accuracy results indicate that CoPro produced accurate context values with 95.05% of all values collected in both experiments determined as accurate based on the statistical method described in Section 5.5.2.

Trustworthiness was evaluated for all low/high-level context values produced by CoPro in the third and fourth experiments conducted as part of the evaluation process of CoPro. A

trustworthiness value was assigned to all context values to indicate the reliability of the context provider that produced the context. This trustworthiness value was also re-calculated based on the completeness QoC metric, which only applied to those contexts with a completeness metric (i.e. location and network). Trustworthiness results are presented in Table 5.20, which denote the occurrences of trustworthiness values in particular ranges. These ranges are 0.6 (incl.) to 0.9 and 0.9 (incl.) to 1 (incl.).

Table 5.20: Pivot table of trustworthiness results of the third and fourth experiments

	Trustworthiness			
	3rd Experiment		**4th Experiment**	
Scenario	**0.6 - 0.9**	**0.9 - 1**	**0.6 - 0.9**	**0.9 - 1**
OIH	2982	10790	3002	10788
OIP	3582	10792	2987	10784
IIH	2990	10783	2990	10783
IIP	3277	10788	3556	10781
Total	12831	43153	12535	43136

Both experiments produced similar trustworthiness results with the third experiment reporting the following results per range: 22.92% (first) and 77.08% (second). The fourth experiment reported the following results per range: 22.52% (first) and 77.48% (second). The similarity between trustworthiness results was due to the fact that no changes were made between the third and fourth experiments. The overall trustworthiness results of CoPro for both experiments highlighted that 77.28% of all context produced can be considered trustworthy as they had a trustworthiness value of between 90 to 100%.

The only specific QoC metric that was used to evaluate the inferred contexts was the certainty QoC metric. Certainty was calculated using freshness, completeness, request for context made and replies for those context requests, as described in Section 5.5.2. The certainty results for both experiments are presented in a consolidated Pivot table shown in Table 5.21. These results were categorized into the following ranges, 0 (incl.) to 0.5, 0.5 (incl.) to 0.9 and 0.9 (incl.) to 1 (incl.).

Table 5.21: Pivot table of certainty results for inferred context of the third and fourth experiments

	Certainty					
	3rd Experiment			4th Experiment		
Scenario	**0 - 0.5**	**0.5 - 0.9**	**0.9 - 1**	**0 - 0.5**	**0.5 - 0.9**	**0.9 - 1**
OIH	0	0	0	0	3	4497
OIP	0	0	0	4	24	4472
IIH	0	0	0	1	32	4467
IIP	0	0	0	4	127	4369
Total	0	0	0	9	186	17805

The results of the third experiment indicated that all the certainty values were reported as 0. By analysing the values obtained from the third experiment and implementation of the certainty QoC metric algorithm, the problem causing no certainty values to be reported was identified. This problem was a simple division problem when trying to divide the requests and replies, which were specified as integers, which resulted in no remainder being reported. The certainty algorithm was changed specifying the requests and replies as doubles in order to correctly calculate the certainty value for each inferred context.

However, when running minor tests the certainty values reported were all below 50%, which indicated there was another issue in calculating the certainty. Upon further investigation the fact that certainty was dependent on the freshness value being greater than 0 in order to signify that a request had been replied to was identified as the issue. As discussed before the freshness QoC metric was improved after the third experiment was conducted and results were analysed. The fourth experiment showed significant improvements in the certainty values of all the inferred contexts. Table 5.21 highlights these improvements by denoting the occurrences of certainty values within the specified ranges consisting of 98.92% of all certainty values within the 90 to 100% range. This result of 98.92% represented the certainty of inferred contexts produced by CoPro.

The quality results presented highlighted the reliability of the context data produced by CoPro in supporting context awareness. The objective evaluation metrics used to evaluate the quality of context produced by CoPro were derived from the QoC metrics in Section 4.3.2. The evaluation metrics determined the quality of the context produced in the third and fourth experiments performed in terms of reliability and usefulness. The analysis of the last two

experiments demonstrated that CoPro produced quality context in terms of low/high-level context as well as inferred context.

5.6.3 Efficacy Results

The efficacy results indicated the capability of CoPro when performing context recognition. Identifying the extent to which CoPro dealt with the limitations of the mobile computing platform validated how capable CoPro was in supporting context awareness.

5.6.3.1 Capability

The results that indicated the capability of CoPro in supporting context awareness were identified by analysing the low/high-level and inferred data produced from all four experiments that were conducted. CoPro was tested in four evaluation scenarios consisting of three test runs per scenario as described in Section 5.5.4. This analysis focused on the process of dealing with limitations of mobile computing in terms of provisioning for limited battery power, limited processing power and dealing with the absence of information. The situations in which CoPro provisioned for the limitations of mobile computing platform were identified.

The first limitation that CoPro needed to consider was limited battery power of a mobile device (Lee, Min, Ju, Pushp & Song 2011). As a mobile device is entirely reliant on battery power, the less battery used the longer the device can be used without needing to recharge it. A situation in which CoPro provisioned to use less battery power was when it dealt with updating the location context. Location context is concerned with the current location of the user. Updating of the user's current location frequently would have a highly negative effect on the battery life of the mobile device. CoPro was thus designed to only update the user's location when they are moving (i.e. not still). An illustration of this can be seen by analysing the samples of data from the first run of the second experiment (OIP) in Table 5.22 below.

Table 5.22: Values for activity and location for first run of second experiment (OIP)

Log	Activity	Location
1	null	At Home 40.0m
3	Tilting. 100%:	At Home 40.0m
4	Tilting. 100%:	At Home 28.28m
12	Tilting. 100%:	At Home 23.09m
29	Standing still. 85%:	At Home 23.09m
30	Standing still. 85%:	At Home 23.09m

The initial location with a confidence of 40m was obtained before a value for activity was reported as seen in Table 5.22. At the 3rd second an initial value for activity was reported (i.e. Tilting 100%) with the location reporting the same initial value. A second after the first initial value for activity was obtained, it was clear that the location confidence had changed from 40m to 28.28m. This change indicates that the location context was updated due to the activity reporting a value of "Tilting". At the 12th second an additional location update was received as indicated by the improved confidence of 28.28m to 23.09m, with activity still reporting a value of "Tilting". This additional location update confirms that when the device was "Tilting" periodic location updates were requested and received. At the 29th second activity reported a new value of "Still 85%" with the location reporting the same value. At the 30th second, the same value for both the activity and location was reported. The same value being reported at the 30th second indicates that no location update was requested when activity was indicated as "Still 85%". This result shows that when activity was not indicated as "Still", location updates were requested and when activity was reported "Still", no location updates were requested. This result therefore demonstrates that CoPro dynamically requested location updates only when needed (i.e. when not "Still"), which would preserve the mobile device's battery life by not making unnecessary location updates.

The second limitation that CoPro needed to consider was limited processing power of a mobile device (Lee *et al*. 2011). This was an important provision to plan for as CoPro would need to perform many tasks simultaneously to support context awareness. These tasks as identified in Section 2.1.1, included acquiring, monitoring, filtering, storing, representing and interpreting context. In order to support all these task of context awareness simultaneously, CoPro made use of a thread pool to manage the execution of each task. A thread pool was used as it allowed individual tasks to execute in multiple threads that were available on the mobile device (i.e. Samsung S4). These individual tasks were added to a queue until such time that a thread was available to process a new task, which would then take the next task to be processed from the queue and execute it. This process of fetching new tasks from the queue to execute, continues until all tasks have been completed. The thread pool used in CoPro had to be managed correctly to avoid increased resource usage. One of the main challenges was to effectively create and destroy new tasks (Garg & Sharapov 2002).

Problems that could have affected the performance of CoPro are highlighted below:

- Creating too many task threads would have wasted resources as well as the time for creating unused task threads.
- Destroying too many task threads would have caused time to be wasted in creating new task threads again.
- Creating task threads too slowly would have resulted in poor performance (lengthy wait times)
- Destroying threads too slowly would have exhausted resource usage that could have been used for other tasks.

The results highlighted in Section 5.6.1 and 5.6.2 demonstrated that the thread pool implementation in CoPro successfully managed all the tasks involved in supporting context awareness simultaneously.

The third and final limitation that CoPro needed to make provision for was the absence of information, which was an existing issue highlighted in Section 2.5. A situation in which CoPro dealt with a lack of information is presented in Table 5.23.

Table 5.23: Values for temperature, humidity and weather for the third run of fourth experiment (OIH)

Log	Temperature	Humidity	Weather
1	20.0°C	52%	broken clouds
2	Very hot 31.73°C	Low 28.73%	broken clouds

As highlighted in Table 5.23, the values for temperature and humidity at Log 1 (i.e. after 1 second elapsed) and Log 2 (i.e. after 2 seconds elapsed) are different. This result is significant as it indicates that at Log 1 no values were reported for both the temperature and humidity context. However, because a value was obtained for the weather at Log 1, both temperature and humidity were assigned values. The values assigned for both the temperature and humidity context at Log 1 came from the weather context obtained, which not only reported the weather condition but also the related temperature and humidity values. The provision demonstrates that CoPro successfully dealt with the lack of information from sensors (i.e. temperature and humidity) with other sources of information such as a web service (i.e. weather).

Another noticeable capability of CoPro, which can be seen as dealing with the absence of information, was provisioning for ambiguous context. This provisioning was highlighted when CoPro was able to enhance the precision of the location context by combining it with preferences. This enhancement of the location context is evident in Table 5.22, whereby the location values are reported as "At Home", which signifies that the location address produced by the GPS was matched with the location preferences.

5.6.4 Discussion

The main goal of the evaluation was to determine the feasibility of the proposed model by evaluating the extent to which CoPro supports context awareness. The three sub-goals formed part of the main goal and together were used to address the fifth research question identified in Chapter 1. The sub-goals specifically focused on the effectiveness, reliability and capability of the prototype, CoPro.

From the utility results, it can be concluded that CoPro can effectively produce current context information, detect changes in context information and enhance context information by using preferences. This conclusion is supported by the extensive analysis of the context information described in Section 5.6.1. This analysis showed that frequent and consistent context values were produced in three of the four experiments that were conducted. The results presented in Table 5.6 to Table 5.9 provided further evidence that CoPro can effectively detect changes in context information. The results in Table 5.5 also showed that context information such as location can be enhanced with preferences such as being at home instead of only providing an address of the location.

CoPro's ability to produce quality context information was validated by the quality results presented in Section 5.6.2. These quality results showed that the freshness of the context values produced was improved by 25.26% with the additional improvements made to the design and implementation of CoPro. The completeness results demonstrated that 71% of all the values reported for the third and fourth experiment had a completeness value of 1 meeting the completeness requirements.

The reliability results highlighted an improvement in context information reliability of 19.67% for values above 50% reliability from the third to fourth experiment. This

improvement further validates the previous changes made to CoPro, which resulted in the fourth experiment producing 88.13% of its values with a reliability above 50%. The granularity results were also enhanced by the experiments from 47.68% to 86.86% for values with a granularity between 90 to 100%.

The confidence interval values for activity were reasonably accurate with 98.63% of them indicated with a confidence of above 90%. Although 41.2% of the location values reported had a confidence of less than 20m, most of the location values were correctly reported and represented the true value of location (i.e. At Home). Overall the accuracy results indicated that 95.05% of the values produced by CoPro from both the third and fourth experiments were considered as accurate based on the statistical method described in Section 5.5.2.

The results for the trustworthiness evaluation metric indicated that 77.28% of the overall context produced by CoPro in the last two experiments were between 90 to 100%. This result highlights that the context providers that produced 77.28% of the context values for CoPro in those two experiments can be considered to be trustworthy. Lastly the certainty results showed that 98.92% of the inferred contexts determined in the fourth experiment had a certainty of between 90 to 100%.

CoPro can successfully deal with the mobile computing platform limitations identified in Section 5.6.3.1. The efficacy results provided evidence to support this statement, which highlighted CoPro's ability to only request location updates when the device is moving to preserve the life of the mobile device's battery. These results also emphasized CoPro's parallel processing pattern with the use of a thread pool to efficiently perform the tasks needed as identified in Section 2.1.1 to support context awareness. Finally the efficacy results also showed the capability of CoPro in using other sources of data (i.e. web services) to compensate when there was lack of information provided from the preferred context providers (i.e. sensors).

Overall these results addressed the main goal as well as the sub-goals identified in Section 5.3, which were aligned with the DSR evaluation activity of the design cycle. As a result, these results also rigorously demonstrated the utility, quality and efficacy of the prototype, CoPro, in supporting context awareness.

5.7 Design Implications and Recommendations

Both literature chapters (Chapters 2 and 3) concluded with requirements, which were identified in order for a context-aware model to support context awareness in mobile applications. These requirements can be compared with the results of the evaluations conducted consisting of four experiments to identify design recommendations for a context-aware model.

The following section will first compare the extent to which the proposed model implemented as the CoPro prototype supported the requirements identified in Chapters 2 and 3. This section will then compare the requirements with the results of the evaluation experiments discussed in Section 5.6 to determine the final design recommendations for a context-aware model to support context awareness in mobile applications.

5.7.1 Support for Requirements

The requirements identified in Chapter 2 highlighted the existing issues in context awareness. The extent to which CoPro supported these issues are described in Table 5.24 for each of the existing issues identified.

Table 5.24: Extent of support for existing issues in context awareness

Design Implication	Supported?	Extent of Support
1. Sensor-based Context Recognition	Yes	CoPro successfully performed sensor-based as well as non-sensor-based context recognition. The available inputs (i.e. sensors) on the device were detected at run-time before commencing the context recognition process.
2. Activity Recognition	Yes	With the use of the Google Activity Recognition API, CoPro was able to detect six different physical activities. CoPro could further determine several inferred activities by using multiple inputs.
3. Indoor Location Awareness	Partially	CoPro was able to detect whether the device was indoor or outdoor. However, detecting room level precision could not be achieved as this is still a major challenge of location awareness.

4. Automated Context Situation Prediction	Yes	CoPro automated the process of determining inferred context at run time by combining several low/high-level contexts.
5. Context Ambiguity	Yes	Dealing with context ambiguity was highlighted when CoPro made use of available weather data to provide values for temperature and humidity. Use of location preferences also enhanced the precision of location values.
6. Appropriate Storage	Partially	As the current context needed to be as fresh as possible, the context was stored within the application for further use. Improvements could involve storing a context history on a remote server or cloud computing platform.
7. User Control and Automation	Yes	Context recognition processes needed to be automated as much as possible, however by allowing preferences to be set enabled a balance between user control and automation.

The requirements identified in Chapter 3 described the shortcomings in existing context-aware models. As the CoPro prototype was based on the proposed model, the extent to which CoPro supported these shortcomings are highlighted in Table 5.25 for each shortcoming identified.

Table 5.25: Extent of support for shortcomings in existing context-aware models

Design Implication	**Supported?**	**Extent of Support**
1. Extract High-Level Context	Yes	CoPro could not only extract high-level context from low-level context (raw data) but also combined this high-level context to produced inferred context.
2. Optimisation Support for Continuous Sensing and Processing	Yes	Provisions for continuous sensing and processing of context were made by CoPro by using a thread pool to manage context related tasks.
3. M-health Context	Partially	As the proposed model focused on obtaining context with only one device, CoPro incorporated a set of health preferences. Bodily sensor readings were not detected as this would have required additional wearable devices to be used.

4. Dimension-based Context Modelling	Yes	The proposed model categorized and modelled all context inputs into four key dimensions including physical, user-activity, health and user preferences.
5. Personalized Context	Yes	The use of user preferences allowed for contexts such as location to be tailored (i.e. location value of "At Home").
6. Dynamic Context	Yes	CoPro considered both the dynamic and static nature of context by using streaming sensor data as well as non-steaming data as input sources

5.7.2 Final Design Recommendations

The final design recommendations are discussed in this section. The results of the iterative evaluation experiments validated all of the requirements identified in Chapters 2 and 3. As a result, the evaluation results and requirements were subsequently used to derive the design recommendations. These design recommendations provide valuable insight to future researchers when designing a model to support context awareness in mobile applications.

The final design recommendations are:

- When trying to develop a model to support context awareness, the use of an existing model that is the most suitable based on the requirements can form a strong starting point. The existing model would have already solved certain issues that would possibly have to be solved again if one chose to develop a context-aware model from scratch.
- Using multiple-input sources when trying to recognize context can help alleviate issues such as when there is an absence of information (an example can be viewed in Section 5.6.3.1).
- Context is multi-dimensional and should be considered in both its static and dynamic form (i.e. steady context like gender vs. changing context such as physical activity).
- In terms of the quality of context, the objective and subjective view of the context quality needs to be considered as each evaluate different aspects of context. (i.e. perceived value versus objective value).
- Considering the limitations of the mobile computing platform is a must when designing for mobile devices. Not considering these limitations could lead to poor performance such as draining the device's battery life unnecessarily.

- Extracting high-level abstract meanings from low-level context data can provide more useful insight about the context, instead of merely reporting the absolute values (i.e. a value of 100 lux could be better represented as "Normal" for the ambient light level).
- Tailoring produced context preferences allows for some user control over the automated context recognition process. This personalization of context can improve the adoption and acceptance rate of context-aware provisions in mobile applications.

5.8 Conclusion

This chapter addressed the evaluation phase of the DSR methodology and the fourth research question identified in Chapter 1: "*How effective, reliable and capable is the proposed context-aware model and to what extent does it support context awareness in mobile applications?"* The evaluation of the proposed model was noted as a core activity of the Design Cycle in DSR. Experimental evaluation methods including controlled experiments and simulation were used to demonstrate well-executed evaluation methods, as proposed by (Hevner *et al*. 2004). The experimental evaluations were conducted in an iterative evaluate and re-design process as this is an important characteristic of DSR. The experimental evaluation methods rigorously demonstrated the utility, quality and efficacy of CoPro.

The chapter first identified evaluation techniques, which could be used to evaluate CoPro. The goals of the evaluation were then described. A suitable evaluation technique was then selected and motivated based on the evaluation goals. The experimental design was then described in detail, which included the evaluation objectives, evaluation metrics, evaluation instruments and evaluation procedure.

The analysis of results as well as the design improvements for each repeated experiment were presented. The utility results (Section 5.6.1) highlighted that CoPro was effective in not only obtaining initial context but also at detecting changes in the current context and tailoring context where applicable with preferences. The quality results (Section 5.6.2) demonstrated that the quality of context produced by CoPro was of high-quality and reliable. The efficacy results showed that CoPro dealt with the limitations of the mobile computing platform by making provisions for limited battery power, limited processing power and absence of information.

Lastly, design implications and design recommendations were discussed to conclude the chapter. CoPro supported all the design implications that were identified from Chapters 2 and 3. Based on these design implications and the results obtained from the evaluation experiments, design recommendations were made for the benefit of future researchers in this area of research.

The final chapter, Chapter 6, concludes this dissertation by providing an overview of the research conducted. The final chapter will highlight the contributions of this research conducted using a DSR methodology in order to add value to the DSR knowledge base as highlighted in Section 1.4.3. Future work for continuation of this research will also be presented in the final chapter. The concluding chapter will complete the DSR process by communicating the findings of this research to the appropriate audience.

Chapter 6: Conclusions

6.1 Introduction

The main objective of this research was to develop a context-aware model to support the context awareness needs of mobile applications. The Design Science Research methodology was used during this development. Implementation of the context-aware model as a prototype formed part of the main objective. The prototype, named CoPro, was implemented for the purpose of assessing the feasibility of the proposed context-aware model in order for it to be considered a viable DSR artefact. Other deliverables of this research included the context-aware model, the evaluation of the prototype and the final design recommendations highlighted in Chapter 5.

This chapter concludes this dissertation by presenting the conclusions and contributions of the research. Conclusions will be inferred about whether the model can facilitate context awareness in mobile applications using multiple inputs. Contributions of this research in terms of theory and practice will be identified. The initial goals of the research will be reviewed to determine whether the research met its planned objectives. A summary of the problems and limitations encountered in conducting this research will be discussed. Lastly, future research will be identified to conclude the chapter.

6.2 Achievement of Research Objectives

This research has shown that a context-aware model can be developed to support the context awareness needs of mobile applications. The implementation of the proposed model can be used to support context awareness in mobile applications and enable mobile applications to take advantage of this valuable information. Context-aware mobile applications could assist in time-critical situations, such as emergency healthcare and location-based services as identified in Section 1.1 in Chapter 1.

The main research objective of this research was to develop a context-aware model for mobile applications using multiple input sources. A discussion of how this research achieved this objective is highlighted in this section. In order to achieve the main objective, several sub-objectives were identified in Section 1.4.2 in Chapter 1, which are shown below:

RO 1. To identify the existing problems and requirements with context awareness in mobile applications (Chapter 2).

RO 2. To identify the existing problems with context awareness solutions that relate to mobile applications (Chapter 3)

RO 3. To develop a context-aware model using multiple input techniques and implement this model into a prototype (Chapter 4).

RO 4. To evaluate the utility, quality and efficacy of the prototype developed based on the proposed model (Chapter 5).

The Relevance Cycle of DSR initiated this research allowing for opportunities and problems to be identified. The problem identified for this research was that mobile applications do not use multiple data inputs to accurately determine context and therefore lack context awareness. This problem is in line with the problem relevance guideline highlighted in Section 1.4.3, as this problem can be applicable to mobile applications created by businesses.

Once the main problem of this research was identified, the Rigor Cycle of DSR allowed for consideration of past knowledge in the form of grounding theories and methods along with domain experience and expertise. The Relevance and Rigor Cycle allowed for Research Objectives 1 and 2 (RO 1 and RO 2) to be addressed by conducting literature studies on the two knowledge bases of this research, namely context awareness and mobile computing. These two literature studies were addressed in Chapter 2 and Chapter 3, which addressed the sub-objectives RO 1 and RO 2. Chapter 2 highlighted the requirements involved when dealing with context awareness as well as the existing issues with context recognition. Chapter 3 discussed context awareness in terms of mobile device, medical health (m-health) as a potential problem domain and problems with existing models that deal with context awareness. Chapter 2 achieved RO 1 by helping to identify requirements that would not only be used to design the proposed model but to also match these requirements to the existing context models identified in Chapter 3. This matching of the findings from Chapter 2 with the findings of Chapter 3 allowed for RO 2 to be achieved as an existing model, which was considered the most complete was used as a basis for the proposed model.

The requirements from successfully achieving RO 1 and the selected existing model as a basis from successfully achieving RO 2 allowed for the third research objective (RO 3) to be addressed. RO 3 was addressed in Chapter 4, which used the deliverables of both RO 1 and

RO 2 to design the proposed model and implement the CoPro prototype based on this model. Chapter 4 embodied the central Design Cycle of DSR, which supported the building and evaluating of the design artefacts. The Design Cycle allowed for an iterative process to be followed in which generated feedback was used to further refine the design of the artefact. After several iterations, the designed model and implemented prototype, CoPro was ready to be evaluated achieving RO 3. Although this build and evaluate process was followed during the initial design of the context-aware model and initial implementation of CoPro, this process is meant to be used more formally in the evaluation of the design artefact. This notion is supported by a characteristic of DSR whereby re-evaluations are conducted to refine the design artefact until a final design artefact is produced. Therefore, the build and evaluate process was also followed during the DSR evaluation phase through performing several evaluation experiments that addressed the fourth and final research objective (RO 4).

To achieve the fourth research objective (RO 4), a total of four experiments were conducted in four evaluation scenarios highlighted in Section 5.5. These experiments evaluated the utility (effectiveness), quality (reliability) and efficacy (capability) of CoPro to determine if the main objective of this research was met. The results of these four experiments highlighted in Section 5.6, concluded that CoPro was indeed effective in producing, detecting and using context with preferences. These results in terms of quality of context also demonstrated the 88.13% of values in the fourth experiment had a reliability of above 50%. The results also highlighted that all the context values that were evaluated in terms of accuracy in both the third and fourth experiments 95.05% of them were determined as accurate using a statistical method. The final results of the repeated evaluation experiments successfully demonstrated the capability of CoPro in dealing with the limitations of the mobile computing platform including, limited battery power, limited processing power and missing information.

The results of all the experiments and particularly the fourth experiment validated that the design decisions made for producing the final prototype were correct as CoPro effectively supported the context awareness process that would be used in mobile applications. Therefore, the evaluation results demonstrated the achievement of all the sub-objectives and as a result, the main objective was achieved, which was to develop a context-aware model for mobile applications using multiple input sources.

The requirements identified in Chapters 2 and 3 were also validated by the results of the evaluation of CoPro and the design of the proposed model. Design recommendations for future researchers interested in designing and implementing a context-aware model were also provided. These design recommendations form part of the contributions of this research and will be described in the next section.

This section has demonstrated the successful achievement of all the research objectives identified forth for this research. The contributions of this research are described in the next section in term of theoretical and practical contributions.

6.3 Research Contributions

Several research contributions were made by successfully achieving the research objectives. These research contributions were categorized into theoretical and practical contributions. The process of developing and designing the proposed model produced theoretical contributions. The implementation of the proposed model into a prototype, named CoPro and evaluation of CoPro produced practical contributions.

6.3.1 Theoretical Contribution

The first theoretical contribution includes the requirements of context awareness in terms of existing issues (Section 2.5) and possible solutions (Section 2.6) to address these problems. These requirements provide further understanding of context awareness and the issues that exist in this research area. The context awareness requirements can thus be used as criteria to assist future researchers when forming solutions to address the problems in context awareness. These requirements represent a significant theoretical contribution in the enhancement of context awareness research.

The second theoretical contribution was made by designing the proposed model based on the highlighted requirements in Chapter 2 and the existing model selected in Chapter 3. The proposed model was designed by taking both the context awareness requirements and the shortcomings of the existing model into consideration. The final design of the proposed model was described in Section 4.2.2 and forms another theoretical contribution in terms of modelling of personal user context.

The third theoretical contribution is related to the second theoretical contribution, namely the design of the proposed model. An underlying context awareness architecture (Section 2.1.1) was adapted and highlighted in Chapter 4 (Section 4.2.3), to form a solid foundation in assisting the proposed model in supporting context awareness in mobile applications. The context awareness architecture assisted in converting the proposed model into a prototype called CoPro. This architecture provides valuable insight into the underlying layers needed to support context-aware applications from the initial sourcing of multiple low-level inputs. This architecture therefore is another significant theoretical contribution.

The fourth and final theoretical contribution are the design recommendations for developing an effective context-aware model described in Section 5.7.2. These design recommendations were derived from the design implications (i.e. context awareness and model requirements) and evaluation results, which were matched in Section 5.7.1. These design recommendations provide valuable knowledge and insight into successfully developing a context-aware model, which were validated by the results of the repeated experiments. Therefore, the set of design recommendations is the final noteworthy theoretical contribution of this research.

6.3.2 Practical Contribution

The first practical contribution of this research is the CoPro prototype that was developed based on the proposed model (Section 5.7.2) to support the context awareness needs in mobile applications. CoPro performed all of the complex tasks involved in both the lower and middle layers of the context awareness architecture. Implementing each of the components of these two layers (i.e. context gathering and pre-processing, and middleware) form part of this contribution. The second part of this contribution was using the proposed model to model the multiple-input sources into the identified context dimensions, including physical, user-activity, health and user-preferences. The combined prototype, CoPro, forms the first substantial practical contribution of this research.

The second practical contribution are the evaluation metrics (Section 5.5.2) that were developed and implemented in the context recognition process within CoPro, and also used for the multiple evaluations conducted. These evaluation metrics were derived from QoC metrics identified in literature (Section 4.3.2). Since only the objective QoC metrics were considered, additional QoC metrics were identified and developed to be used to ensure a

rigorous evaluation. The results of the third evaluation experiment revealed that some QoC metrics were not producing accurate values such, as the freshness metric. This highlighted that those QoC metrics defined in literature were not validated in a practical manner to see if they actually work. These QoC metrics were thus reworked to produce new QoC metrics to be used for evaluation of CoPro. The results of the fourth experiment confirmed the accuracy and correctness of the updated QoC metrics. Therefore, all the QoC metrics used for evaluating the feasibility of CoPro form the second practical contribution to the DSR knowledge base (Section 1.4.3).

The third practical contribution is the evaluation design and the results of this evaluation. The evaluation design is a noteworthy part of this contribution as there is limited DSR evaluation literature on how an experimental DSR evaluation should be conducted at an operational level. The evaluation design can therefore help future researchers in conducting DSR evaluations using experimental design. The evaluation results that were obtained from the four evaluation experiments performed to evaluate CoPro are also part of this contribution. These results rigorously demonstrated the effectiveness, quality and efficacy of the CoPro prototype and the proposed model. These conclusions show that both CoPro and the proposed context-aware model support the context awareness needs in mobile applications and are therefore viable artefacts.

6.4 Encountered Problems and Limitations

Several problems were encountered during the design, implementation and evaluation cycles of this research. One of these problems involved dealing with the multiple input sources reported from different context providers (i.e. sensors, web services, calendar). During the proposal of this research topic, this aspect of the research was considered reasonable. However, when actually conducting the research this task was identified as inherently complex. Information provided from the sensors, web services and calendar all needed to be handled and processed individually. For example, the sound values obtained from the mobile device's microphone had to be calibrated as highlighted in Chapter 4 (Section 4.3.2). This calibration involved calibrating the sound values with a sound level meter in order to accurately report the actual ambient sound level. This calibration was needed to report valid sound level values as the mobile device's microphone did not report the actual sound level.

Another problem encountered was when some QoC metrics identified from literature were used during the evaluation, but did not produce valid results as discussed in Chapter 5 (Section 5.6.2.1). For example the freshness QoC metric did not report expected values for the freshness of the context data. The time period of the freshness QoC formula had to be calculated in a completely new way compared to its initial definition in order to produce valid freshness values. A limitation also related to the evaluation was the limited research available for guiding and performing the experimental design evaluation at an operational research level.

The last problem that was faced occurred after the logging of the evaluation context data (Section 5.5.4). There was a problem in extracting the logged data from the mobile device. In general, it should be possible to download the logged data directly from the mobile device; however, this was not the case. In order to solve this problem, the logged data was emailed for further analysis by the evaluator (author) after being transferred into CSV files.

The next section will highlight potential future research that could be conducted for the purposes of making further significant developments in this research area.

6.5 Future Research

This section concludes this chapter and this dissertation by highlighting future research that can potentially improve the developments made by this research. CoPro and the proposed model were designed to effectively handle at least sixteen different input sources (excluding preferences). However, because of the robust design of the proposed model, additional input sources could be easily included to provide even more valuable context information. An example could be to add geomagnetic field data from the magnetic field sensor. Using the magnetic field sensor with some algorithms could assist in determining in which direction the user is facing, such as magnetic north. With new sensors being constantly added to mobile devices, such as the Samsung S5 with its heart rate monitor sensor, there is real potential for future research in this area.

It would also be valuable to test the proposed model with users by generating scenarios in which the users perform specific tasks and CoPro would then provide information based on the users' context. The users' opinions and their perceived value of the context-aware services

could then be captured via user surveys and questionnaires. These tests could be performed in two ways, either controlled by using pre-determined tasks or in a more natural environment and allowing the user to use the system freely over a period of time. This would also allow the subjective QoC metrics identified such as the validity and significance of context, to be used according to the user requirements. Using these subjective QoC metrics would provide a higher level of personalization of context, which would support the adoption of context-aware services.

Another possible avenue for extending this research could be to convert CoPro into an application programming interface (API) to allow mobile application developers to use this as a context-aware service. As CoPro can reliably and accurately perform context recognition for low-level and high-level as well as inferred context, mobile application developers could use the CoPro API to develop context-aware applications. Assessing how the developers use the API and whether they find it simple and easy to use and useful could represent a significant contribution as part of future research in this area.

List of References

Agarwal, S. & Lau, C.T., 2010. Remote health monitoring using mobile phones and Web services. *Telemedicine journal and e-health*, 16(5), pp.603–607. Available at: Available at: http://www.ncbi.nlm.nih.gov/pubmed/20575728.

AHIMA, 2013. What is a Personal Health Record (PHR)? Available at: http://www.myphr.com/StartaPHR/what_is_a_phr.aspx [Accessed July 20, 2013].

Álamo, P.C. & Vilariño, A.B.L., 2012. The Context Manager: Personalized Information and Services in Mobile Environments. *UBICOMM 2012*, p.29. Available at: http://www.slideshare.net/p_curiel/ubicomm-2012-the-context-manager-personalized-information-and-services-in-mobile-environments [Accessed June 5, 2013].

Alidin, A.A. & Crestani, F., 2012. Context acquisition in just-in-time mobile information retrieval. *2012 International Conference on Information Retrieval & Knowledge Management*, pp.203–207. Available at: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6205002.

Antila, V., Polet, J. & Sarjanoja, A., 2011. ContextCapture: exploring the usage of context-based awareness cues in informal information sharing. *Proceedings of the 15th International Academic MindTrek Conference: Envisioning Future Media Environments*, pp.269–275.

Baldauf, M., Dustdar, S. & Rosenberg, F., 2007. A survey on context-aware systems. *International Journal of Ad Hoc and Ubiquitous Computing*, 2(4), p.263.

Bao, T., Cao, H., Chen, E., Tian, J. & Xiong, H., 2010. An Unsupervised Approach to Modeling Personalized Contexts of Mobile Users. *2010 IEEE International Conference on Data Mining*, pp.38–47. Available at: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=5693957 [Accessed October 31, 2013].

Barkhuus, L. & Dey, A., 2003. Is Context-Aware Computing Taking Control Away from the User ? Three Levels of Interactivity Examined. *UbiComp 2003: Ubiquitous Computing*, pp.149–156.

Bessi, M. & Bruni, L., 2009. A survey about context-aware middleware. , p.45. Available at: http://www.slideshare.net/LeoBruni/a-survey-about-contextaware-middleware [Accessed June 25, 2013].

Bilyeu, D.K., Hardy, D.R., Katz, M.R., Kennelly, B.B. & Warshawsky, M.J., 2009. The Unsustainable Cost of Health Care. , (September). Available at: http://www.ssab.gov/documents/TheUnsustainableCostofHealthCare_graphics.pdf.

Bolchini, C., Curino, C.A., Quintarelli, E., Schreiber, F.A. & Tanca, L., 2007. A Data-oriented Survey of Context Models. *ACM SIGMOD Record*, 36(4), pp.19–26.

Boytsov, A. & Zaslavsky, A., 2011. From Sensory Data to Situation Awareness: Enhanced Context Spaces Theory Approach. *2011 IEEE Ninth International Conference on Dependable, Autonomic and Secure Computing*, pp.207–214. Available at: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6119081 [Accessed February 23, 2014].

Brander, W. & Wesson, J., 2007. Implementing Adaptive Mobile Notification Services : A Model Based Approach. , pp.89–101.

Bricon-Souf, N. & Newman, C.R., 2007. Context awareness in health care: a review. *International journal of medical informatics*, 76(1), pp.2–12. Available at: http://www.ncbi.nlm.nih.gov/pubmed/16488663 [Accessed March 16, 2013].

Broens, T., Halteren, A. Van, Sinderen, M. Van & Wac, K., 2007. Towards an application framework for context-aware m-health applications. *International Journal of Internet Protocol Technology*, 2(2), p.109.

Buchholz, T., Küpper, A. & Schiffers, M., 2003. Quality of Context : What It Is And Why We Need It. *Proceedings of the 10th Workshop of the OpenView University Association: OVUA'03*, pp.1–14.

Chang, Y.-L., Barrenechea, E. & Alencar, P., 2010. Dynamic user-centric mobile context model. *2010 Fifth International Conference on Digital Information Management (ICDIM)*, pp.442–447. Available at: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=5662575.

Chen, H., Finin, T. & Joshi, A., 2003. An ontology for context-aware pervasive computing environments. *The Knowledge Engineering Review*, 18(3), pp.197–207.

Cholakian, A., 2009. web services - API vs Webservice - Stack Overflow. Available at: http://stackoverflow.com/questions/808421/api-vs-webservice [Accessed June 17, 2014].

Choudhury, T., Borriello, G., Consolvo, S., Haehnel, D., Harrison, B., Hemingway, B., Hightower, J., Klasnja, P., Koscher, K., LaMarca, A., Landay, J.A., LeGrand, L., Lester, J., Rahimi, A., Rea, A. & Wyatt, D., 2008. The mobile sensing platform: An embedded activity recognition system. *IEEE Pervasive Computing*, 7(2), pp.32–41.

Christoph, U., Krempels, K., Stulpnagel, J. Von & Terwelp, C., 2010. Automatic context detection of a mobile user. *International Conference on Wireless Information Networks and Systems (WINSYS)*, pp.1–6.

Clarkson, B., Mase, K. & Pentland, A., 2000. Recognizing user context via wearable sensors. *Digest of Papers. Fourth International Symposium on Wearable Computers*.

Curran, K., 2011. Ambient Intelligence – Context Aware, Pervasive and Making a Difference in a Modern World K. Curran, ed. *Ubiquitous Innovative Applications of Ambient Intelligence: Advances in Smart Systems*, (2011), pp.i–xv. Available at: http://services.igi-global.com/resolvedoi/resolve.aspx?doi=10.4018/978-1-46660-038-6 [Accessed May 17, 2013].

David, L., Endler, M., Barbosa, S.D.J. & Filho, J.V., 2011. Middleware Support for Context-Aware Mobile Applications with Adaptive Multimodal User Interfaces. *2011 Fourth International Conference on Ubi-Media Computing*, pp.106–111. Available at: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=5992054 [Accessed October 31, 2013].

Debes, M., Lewandowska, A. & Seitz, J., 2005. Definition and Implementation of Context Information. *Proceedings of the 2nd Workshop on Positioning, Navigation and Communication & 1st Ultra-Wideband Expert Talk*, pp.63–68. Available at: http://www.wpnc.net/fileadmin/WPNC05/Proceedings/Definition_and_Implementation_of_Context_Information.pdf.

Demeester, P., 2010. Context Aware Services. *2de IBBT Brokerage Event Break-out Sessie*, p.24. Available at: www.slideshare.net/guest3cf4991/5-context-aware-services [Accessed June 5, 2013].

Dey, A.K. & Abowd, G.D., 1999. Towards a Better Understanding of Context and Context-Awareness. *Computing Systems*, 40(3), pp.304–307.

Dey, A.K., Abowd, G.D. & Salber, D., 2001. A Conceptual Framework and a Toolkit for Supporting the Rapid Prototyping of Context- Aware Applications. *Human – Computer Interaction*, 16(2-4), pp.97–166.

Dey, A.K. & Mankoff, J., 2005. Designing mediation for context-aware applications. *ACM Transactions on Computer-Human Interaction*, 12(1), pp.53–80.

Dunwiddie, B., 2014. CSV File Format. Available at: http://www.csvreader.com/csv_format.php [Accessed August 20, 2014].

Eichler, G., Lüke, K. & Reufenheuser, B., 2009. Context information as enhancement for mobile solutions and services. *13th International Conference on Intelligence in Next Generation Networks: Beyond the Bit Pipes, (ICIN 2009)*, pp.1–5.

Elgazzar, K., Aboelfotoh, M., Martin, P. & Hassanein, H.S., 2012. Ubiquitous Health Monitoring Using Mobile Web Services. *Procedia Computer Science*, 10, pp.332–339. Available at: http://linkinghub.elsevier.com/retrieve/pii/S1877050912004012 [Accessed February 28, 2013].

Falchuk, B., Loeb, S. & Panagos, T., 2008. Seamless Mobile Context - Towards High Speed Context-Aware Event-Based Middleware. *2008 5th IEEE Consumer Communications and Networking Conference*, pp.89–90. Available at: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=4446324.

Fausto, M. & Alberto, P., 2010. Context Planning and User Profiling in Mobile Services. *Information Technology Interfaces (ITI)*, pp.301–306.

Filho, J.B., Miron, A.D., Satoh, I., Gensel, J. & Martin, H., 2010. Modeling and Measuring Quality of Context Information in Pervasive Environments. *2010 24th IEEE International Conference on Advanced Information Networking and Applications*, pp.690–697. Available at:

http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=5474806 [Accessed December 10, 2014].

Fisher, J. a & Monahan, T., 2012. Evaluation of real-time location systems in their hospital contexts. *International journal of medical informatics*, 81(10), pp.705–12. Available at: http://www.ncbi.nlm.nih.gov/pubmed/22857790 [Accessed March 4, 2013].

Garg, R.P. & Sharapov, I., 2002. *Techniques for Optimizing Applications: High Performance Computing*, Prentice Hall Professional Technical Reference.

Gehlen, G., Aijaz, F., Sajjad, M. & Walke, B., 2007. A Mobile Context Dissemination Middleware. *Fourth International Conference on Information Technology (ITNG'07)*, pp.155–160. Available at: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=4151675.

Google Inc, 2014a. Environment Sensors | Android Developers. Available at: http://developer.android.com/guide/topics/sensors/sensors_environment.html [Accessed June 18, 2014].

Google Inc, 2014b. Issue 7981: Documentation Enhancement: SensorEvent timestamp. Available at: https://code.google.com/p/android/issues/detail?id=7981 [Accessed June 18, 2014].

Google Inc, 2014c. Jelly Bean | Android Developers. Available at: http://developer.android.com/about/versions/jelly-bean.html#media [Accessed June 17, 2014].

Google Inc, 2014d. Location APIs | Android Developers. Available at: https://developer.android.com/google/play-services/location.html [Accessed June 17, 2004].

Google Inc, 2014e. Location Strategies | Android Developers. Available at: http://developer.android.com/guide/topics/location/strategies.html [Accessed June 17, 2014].

Google Inc, 2014f. Sensor | Android Developers. Available at: http://developer.android.com/reference/android/hardware/Sensor.html#getMinDelay() [Accessed June 18, 2014].

Google Inc, 2014g. SensorEvent | Android Developers. Available at: http://developer.android.com/reference/android/hardware/SensorEvent.html [Accessed June 18, 2014].

Google Inc, 2014h. SensorManager | Android Developers. Available at: http://developer.android.com/reference/android/hardware/SensorManager.html#SENSOR_STATUS_UNRELIABLE [Accessed June 18, 2014].

Google Inc, 2014i. Sensors Overview | Android Developers. Available at: http://developer.android.com/guide/topics/sensors/sensors_overview.html [Accessed June 17, 2014].

Gray, P. & Salber, D., 2010. Modelling and using sensed context information in the design of interactive applications M. R. Little & L. Nigay, eds. *Computer*, 2254(1), pp.317–335.

Guerri, J.C., Antón, A.B., Pajares, A., Monfort, M. & Sánchez, D., 2008. A mobile device application applied to low back disorders. *Multimedia Tools and Applications*, 42(3), pp.317–340. Available at: http://link.springer.com/10.1007/s11042-008-0252-x [Accessed June 7, 2013].

Hardian, B., 2011. Context Awareness in Mobile Computing. *Seminar Riset Tenologi Informasi*, p.29. Available at: http://www.slideshare.net/bhardian/context-awareness-in-mobile-computing [Accessed June 5, 2013].

Hassani, M. & Seidl, T., 2011. Towards a Mobile Health Context Prediction: Sequential Pattern Mining in Multiple Streams. *2011 IEEE 12th International Conference on Mobile Data Management*, pp.55–57. Available at: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6068495 [Accessed October 31, 2013].

Health IT Inc., 2013. Maintain Your Medical Record. Available at: http://healthit.gov/patients-families/maintain-your-medical-record [Accessed July 20, 2013].

Hevner, A.R., 2007. A Three Cycle View of Design Science Research. *Scandinavian Journal of Information Systems*, 19(2), pp.87–92.

Hevner, A.R., March, S.T., Park, J. & Ram, S., 2004. Design Science in Information Systems Research A. R. Hevner & S. Chatterjee, eds. *MIS Quarterly*, 28(1), pp.75–105. Available at: http://www.jstor.org/stable/25148625.

Hong, J., Suh, E. & Kim, S.-J., 2009. Context-aware systems: A literature review and classification. *Expert Systems with Applications*, 36(4), pp.8509–8522. Available at: http://linkinghub.elsevier.com/retrieve/pii/S0957417408007574 [Accessed July 17, 2014].

Huang, D., Liu, W. & Li, X., 2011. A survey on context awareness. *2011 International Conference on Computer Science and Service System (CSSS)*, pp.144–147. Available at: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=5972040.

Kaenampornpan, M.J., 2004. Bringing Context Awareness Together. *Second International Conference on Pervasive Computing*.

Kang, S., Iyengar, S.S., Lee, Y., Min, C., Ju, Y., Park, T., Lee, J. & Song, J., 2010. MobiCon : Mobile Context Monitoring Platform for Sensor-Rich Dynamic Environments. *Communications of the ACM Vol. 55*, p.14.

Kerr, J., Duncan, S., Schipperijn, J. & Schipperjin, J., 2011. Using global positioning systems in health research: a practical approach to data collection and processing. *American journal of preventive medicine*, 41(5), pp.532–40. Available at: http://www.ncbi.nlm.nih.gov/pubmed/22011426 [Accessed March 18, 2013].

Kim, Y. & Lee, K., 2006. A Quality Measurement Method of Context Information in Ubiquitous Environments. *International Conference on Hybrid Information Technology*, 2.

Klasnja, P. & Pratt, W., 2011. Healthcare in the pocket: Mapping the space of mobile-phone health interventions. *Journal of Biomedical Informatics*, 45(1), pp.1–15. Available at: http://www.ncbi.nlm.nih.gov/pubmed/21925288 [Accessed February 28, 2013].

Kosta, E., Pitkänen, O., Niemelä, M. & Kaasinen, E., 2010. Mobile-centric ambient intelligence in Health- and Homecare-anticipating ethical and legal challenges. *Science and Engineering Ethics*, 16(2), pp.303–323.

Lee, K., Lunney, T., Curran, K. & Santos, J., 2009. Ambient Middleware for Context-Awareness (AMiCA). *International Journal of Ambient Computing and Intelligence (IJACI)*, 1(3), pp.66–78.

Lee, M.R. & Chen, T.T., 2009. Trends in Ubiquitous Multimedia Computing. *International Journal of Multimedia and Ubiquitous Engineering*, 4(2), pp.115–124.

Lee, Y., Ju, Y., Min, C., Yu, J. & Song, J., 2012. MobiCon : Mobile Context Monitoring Platform. , (March), pp.116–121. Available at: http://dl.acm.org/citation.cfm?id=2093567&dl=ACM&coll=DL.

Lee, Y., Min, C., Ju, Y., Pushp, S. & Song, J., 2011. A Mobile Context Monitoring Platform for Pervasive Computing Environments. *IEEE International Conference on Digital Ecosystems and Technologies*, pp.345–348.

Lester, J., Choudhury, T. & Borriello, G., 2006. A Practical Approach to Recognizing Physical Activities. *Pervasive Computing*, pp.1–16. Available at: http://www.springerlink.com/index/7048888592382352.pdf.

Liu, C., Zhu, Q., Holroyd, K. a & Seng, E.K., 2011. Status and trends of mobile-health applications for iOS devices: A developer's perspective. *Journal of Systems and Software*, 84(11), pp.2022–2033. Available at: http://linkinghub.elsevier.com/retrieve/pii/S0164121211001610 [Accessed March 6, 2013].

Liu, X. & Karimi, H. a., 2006. Location awareness through trajectory prediction. *Computers, Environment and Urban Systems*, 30(6), pp.741–756. Available at: http://linkinghub.elsevier.com/retrieve/pii/S0198971506000160 [Accessed June 7, 2013].

Lo, C.A., Lin, T., Wang, Y., Tseng, Y., Ko, L. & Kuo, L., 2010. Using Intelligent Mobile Devices for Indoor Wireless Location Tracking , Navigation , and Mobile Augmented Reality.

Lukowicz, P., Nanda, S., Narayanan, V., Albelson, H., Mcguinness, D.L. & Jordan, M.I., 2012. Qualcomm context-awareness symposium sets research agenda for context-aware smartphones. *IEEE Pervasive Computing*, 11(1), pp.76–79.

Malik, N., Mahmud, U. & Javed, Y., 2007. Future Challenges in Context-aware Computing. *Proceedings of the IADIS International Conference*, pp.306–310. Available at: http://www.researchgate.net/publication/221719887_Future_Challenges_in_Context_Aware_Computing/file/9fcfd50541789e2b66.pdf.

Manzoor, A., Truong, H. & Dustdar, S., 2010. Quality of Context : Models and Applications for Context-aware Systems in Pervasive Environments. *The Knowledge Engineering Review*, 29(02), pp.154–170. Available at: http://www.journals.cambridge.org/abstract_S0269888914000034 [Accessed December 4, 2014].

Manzoor, A., Truong, H.L. & Dustdar, S., 2008. On the evaluation of quality of context. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*. pp. 140–153.

March, S.T. & Smith, G.F., 1995. Design and natural science research on information technology. *Decision Support Systems*, 15(4), pp.251–266.

Marcu, M., Ghiata, N. & Cretu, V., 2013. Extracting high-level user context from low-level mobile sensors data. *2013 IEEE 8th International Symposium on Applied Computational Intelligence and Informatics (SACI)*, pp.449–454. Available at: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6609017.

Markus, M.L., Majchrzak, A. & Gasser, L., 2002. A Design Theory for Systems That Support Emergent Knowledge Processes. *MIS Quarterly*, 26(3), pp.179–212. Available at: http://www.jstor.org/stable/4132330.

Martin, D., Lamsfus, C. & Alzua, A., 2011. Mobile Context Data Management Framework. *2011 Fifth FTRA International Conference on Multimedia and Ubiquitous Engineering*, pp.73–78. Available at: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=5992174 [Accessed October 31, 2013].

Matyjarvi, J., 2003. Sensor-based context recognition for mobile applications. *VTT Publications*, (511), pp.1–118. Available at: http://www.scopus.com/scopus/inward/record.url?eid=2-s2.0-0742285269&partner=40&rel=R4.0.0.

Mettler, T., Eurich, M. & Winter, R., 2014. On the Use of Experiments in Design Science Research : A Proposition of an Evaluation Framework Evaluation Framework. *Communications of the Association for Information Systems*, 34(1), pp.223–240.

Mihalic, K., Tscheligi, M. & Unit, U., 2006. Interactional Context for Mobile Applications. *Proceeding of the 20th International Symposium on Human Factors in Telecommunication HFT 2006*.

Milette, G. & Stroud, A., 2012. *Professional Android Sensor Programming*, Available at: http://pdf.th7.cn/down/files/1312/professional_android_sensor_programming.pdf.

Mitchell, M.J., 2011. Context and Bio-Aware Mobile Applications. Available at: http://diginole.lib.fsu.edu/etd/2376/.

Mostefaoui, G.K., Pasquier-Rocha, J. & Brezillon, P., 2004. Context-Aware Computing: A Guide for the Pervasive Computing Community. *The IEEE/ACS International Conference on Pervasive Services*.

Ntawanga, F., Calitz, A.P. & Barnard, L., 2013. An integrated logical context sensor for mobile web applications. *Proceedings of the South African Institute for Computer Scientists and Information Technologists Conference on - SAICSIT '13*, p.320. Available at: http://dl.acm.org/citation.cfm?doid=2513456.2513489.

Oh, Y., Schmidt, A., Woo, W. & Korea, S., 2007. Designing , Developing , and Evaluating Context-Aware Systems. *2007 International Conference on Multimedia and Ubiquitous Engineering (MUE'07)*, pp.1158–1163. Available at: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=4197434.

OpenWeatherMap Inc, 2014. Current weather - OpenWeatherMap. Available at: http://openweathermap.org/current [Accessed June 17, 2014].

Ostrowski, L. & Helfert, M., 2012. Design Science Evaluation – Example of Experimental Design. , 3(9), pp.253–262.

Otebolaku, A.M. & Andrade, M.T., 2013. Recognizing High-Level Contexts from Smartphone Built-In Sensors for Mobile Media Content Recommendation. *2013 IEEE 14th International Conference on Mobile Data Management*, pp.142–147. Available at: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6569080 [Accessed October 31, 2013].

Pantsar-syvaniemi, S., Simula, K. & Ovaska, E., 2010. Context-Awareness in Smart Spaces. *Proceedings - IEEE Symposium on Computers and Communications*, pp.1023–1028.

Pather, D., Wesson, J. & Cowley, L., 2014. A Model for Context Awareness for Mobile Applications using Multiple-Input Sources. *Proceedings of the 2014 Southern Africa Telecommunication Networks and Applications Conference (SATNAC 2014)*.

Peffers, K., Tuunanen, T., Rothenberger, M.A. & Chatterjee, S., 2007. A Design Science Research Methodology for Information Systems Research. *Journal of Management Information Systems*, 24(3), pp.45–78.

Phillips, G., 2014. How To Use An Excel Pivot Table For Data Analysis. Available at: http://www.makeuseof.com/tag/use-excel-pivot-table-data-analysis/ [Accessed August 20, 2014].

PricewaterhouseCoopers' Health Research Institute, 2010. Healthcare Unwired. , p.36. Available at: http://www.pwc.com/il/en/pharmaceuticals/assets/healthcare-unwired.pdf.

Pries-Heje, J., Baskerville, R. & Venable, J.R., 2008. Strategies for Design Science Research Evaluation. In *European Conference on Information Systems (ECIS)*. p. 87.

Rahmati, A., Shepard, C., Tossell, C., Zhong, L. & Kortum, P., 2012. Practical Context Awareness : Measuring and Utilizing the Context Dependency of Mobile Usage. *Technical report 2012-08-31, Rice University*.

Ranganathan, A., Al-Muhtadi, J. & Campbell, R.H., 2004. Reasoning about uncertain contexts in pervasive computing environments. *IEEE Pervasive Computing*, 3(2), pp.62–70.

Ravi, N., Dandekar, N., Mysore, P. & Littman, M.M.L., 2005. Activity recognition from accelerometer data. *Proceedings of the national …*, pp.1541–1546. Available at: http://www.aaai.org/Papers/IAAI/2005/IAAI05-013.

Sam, 2012. android - SensorEvent.timestamp date conversion issue - Stack Overflow. Available at: http://stackoverflow.com/questions/12130666/sensorevent-timestamp-date-conversion-issue [Accessed June 18, 2014].

Santos, A.C., Cardoso, J.M.P., Ferreira, D.R., Diniz, P.C. & Chaínho, P., 2010. Providing user context for mobile and social networking applications. *Pervasive and Mobile Computing*, 6(3), pp.324–341. Available at: http://linkinghub.elsevier.com/retrieve/pii/S1574119210000052 [Accessed November 8, 2014].

Santos, A.C., Tarrataca, L., Cardoso, J.M.P., Ferreira, D.R., Diniz, P.C. & Chainho, P., 2009. Context Inference for Mobile Applications in the UPCASE Project. *Lecture Notes of the Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering*, pp.352–365.

Schmidt, A., Aidoo, K., Takaluoma, A., Tuomela, U., Van Laerhoven, K., Van de Velde, W. & Asante, K., 1999. Advanced Interaction in Context. *Proceedings of First International Symposium on Handheld and Ubiquitous Computing*, 1707, pp.89–101. Available at: http://www.teco.edu/~albrecht/publication/huc99/advanced_interaction_context.pdf.

Schutzberg, A., 2013. Ten Things You Need to Know About Indoor Positioning. Available at: http://www.directionsmag.com/articles/10-things-you-need-to-know-about-indoor-positioning/324602.

Syed, Z. & Finin, T., 2011. Creating and Exploiting a Hybrid Knowledge Base for Linked Data. In *Communications in Computer and Information Science*. pp. 3–21.

Traynor, D., Xie, E. & Curran, K., 2010. Context-Awareness in Ambient Intelligence. *International Journal of Ambient Computing and Intelligence*, 2(1), pp.13–23. Available at: http://services.igi-global.com/resolvedoi/resolve.aspx?doi=10.4018/jaci.2010010102 [Accessed May 17, 2013].

Trifa, V., Guinard, D. & Mayer, S., 2011. Leveraging the Web for a Distributed Location-aware Infrastructure for the Real World. In E. Wilde & C. Pautasso, eds. *REST: From Research to Practice*. Springer New York, pp. 381–400. Available at: http://dx.doi.org/10.1007/978-1-4419-8303-9_17.

Venable, J., 2006. A framework for Design Science Research activities. In M. Khosrow-Pour, ed. *Proceedings of the 2006 Information Resources Management Association*. Idea Group, pp. 184–187. Available at: http://scholar.google.com/scholar?hl=en&btnG=Search&q=intitle:A+Framework+for+Design+Science+Research+Activities#5.

Ventä, L., Isomursu, M., Ahtinen, A. & Ramiah, S., 2008. "My Phone is a Part of My Soul" - How People Bond with Their Mobile Phones. *Proceedings - The 2nd International Conference on Mobile Ubiquitous Computing, Systems, Services and Technologies, (UBICOMM 2008)*, pp.311–317.

Verial, D., 2014. What Is Meant by Cross Tabulation? | The Classroom. Available at: http://classroom.synonym.com/meant-cross-tabulation-6241.html [Accessed August 20, 2014].

Walpole, R., Myers, R., Myers, S. & Ye, K., 2002. *Probability and Statistics for Engineers and Scientists* 7th editio., Pearson Education.

Wolf, H., Herrmann, K. & Rothermel, K., 2010. Robustness in context-aware mobile computing. *2010 IEEE 6th International Conference on Wireless and Mobile Computing, Networking and Communications*, pp.46–53. Available at: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=5645026.

Wu, H., 2003. Sensor Data Fusion for Context-Aware Computing Using Dempster-Shafer Theory. , pp.1–195. Available at: http://portal.acm.org/citation.cfm?id=1023486.

Yang, Y. & Padmanabhan, B., 2005. Evaluation of Online Personalization Systems : A Survey of Evaluation Schemes and a Knowledge-based Approach. , (1), pp.112–122.

Zheng, D., Wang, J. & Kerong, B., 2012. Evaluation of Quality Measure Factors for the Middleware Based Context-Aware Applications. *2012 IEEE/ACIS 11th International Conference on Computer and Information Science*, pp.403–408.

Zhou, P., Zheng, Y., Li, Z., Li, M. & Shen, G., 2012. IODetector : A Generic Service for Indoor Outdoor Detection. *SenSys '12 Proceedings of the 10th ACM Conference on Embedded Network Sensor System*, pp.113–126.

Zhu, J., Chen, P., Pung, H., Oliya, M., Sen, S. & Wong, W., 1992. Coalition : A Platform for Context-Aware Mobile Application Development. *Ubiquitous Computing and Communication Journal*, 6(1), pp.722–735.

Appendix A: T-distribution Table

t Table

cum. prob	$t_{.50}$	$t_{.75}$	$t_{.80}$	$t_{.85}$	$t_{.90}$	$t_{.95}$	$t_{.975}$	$t_{.99}$	$t_{.995}$	$t_{.999}$	$t_{.9995}$
one-tail	**0.50**	**0.25**	**0.20**	**0.15**	**0.10**	**0.05**	**0.025**	**0.01**	**0.005**	**0.001**	**0.0005**
two-tails	**1.00**	**0.50**	**0.40**	**0.30**	**0.20**	**0.10**	**0.05**	**0.02**	**0.01**	**0.002**	**0.001**
df											
1	0.000	1.000	1.376	1.963	3.078	6.314	12.71	31.82	63.66	318.31	636.62
2	0.000	0.816	1.061	1.386	1.886	2.920	4.303	6.965	9.925	22.327	31.599
3	0.000	0.765	0.978	1.250	1.638	2.353	3.182	4.541	5.841	10.215	12.924
4	0.000	0.741	0.941	1.190	1.533	2.132	2.776	3.747	4.604	7.173	8.610
5	0.000	0.727	0.920	1.156	1.476	2.015	2.571	3.365	4.032	5.893	6.869
6	0.000	0.718	0.906	1.134	1.440	1.943	2.447	3.143	3.707	5.208	5.959
7	0.000	0.711	0.896	1.119	1.415	1.895	2.365	2.998	3.499	4.785	5.408
8	0.000	0.706	0.889	1.108	1.397	1.860	2.306	2.896	3.355	4.501	5.041
9	0.000	0.703	0.883	1.100	1.383	1.833	2.262	2.821	3.250	4.297	4.781
10	0.000	0.700	0.879	1.093	1.372	1.812	2.228	2.764	3.169	4.144	4.587
11	0.000	0.697	0.876	1.088	1.363	1.796	2.201	2.718	3.106	4.025	4.437
12	0.000	0.695	0.873	1.083	1.356	1.782	2.179	2.681	3.055	3.930	4.318
13	0.000	0.694	0.870	1.079	1.350	1.771	2.160	2.650	3.012	3.852	4.221
14	0.000	0.692	0.868	1.076	1.345	1.761	2.145	2.624	2.977	3.787	4.140
15	0.000	0.691	0.866	1.074	1.341	1.753	2.131	2.602	2.947	3.733	4.073
16	0.000	0.690	0.865	1.071	1.337	1.746	2.120	2.583	2.921	3.686	4.015
17	0.000	0.689	0.863	1.069	1.333	1.740	2.110	2.567	2.898	3.646	3.965
18	0.000	0.688	0.862	1.067	1.330	1.734	2.101	2.552	2.878	3.610	3.922
19	0.000	0.688	0.861	1.066	1.328	1.729	2.093	2.539	2.861	3.579	3.883
20	0.000	0.687	0.860	1.064	1.325	1.725	2.086	2.528	2.845	3.552	3.850
21	0.000	0.686	0.859	1.063	1.323	1.721	2.080	2.518	2.831	3.527	3.819
22	0.000	0.686	0.858	1.061	1.321	1.717	2.074	2.508	2.819	3.505	3.792
23	0.000	0.685	0.858	1.060	1.319	1.714	2.069	2.500	2.807	3.485	3.768
24	0.000	0.685	0.857	1.059	1.318	1.711	2.064	2.492	2.797	3.467	3.745
25	0.000	0.684	0.856	1.058	1.316	1.708	2.060	2.485	2.787	3.450	3.725
26	0.000	0.684	0.856	1.058	1.315	1.706	2.056	2.479	2.779	3.435	3.707
27	0.000	0.684	0.855	1.057	1.314	1.703	2.052	2.473	2.771	3.421	3.690
28	0.000	0.683	0.855	1.056	1.313	1.701	2.048	2.467	2.763	3.408	3.674
29	0.000	0.683	0.854	1.055	1.311	1.699	2.045	2.462	2.756	3.396	3.659
30	0.000	0.683	0.854	1.055	1.310	1.697	2.042	2.457	2.750	3.385	3.646
40	0.000	0.681	0.851	1.050	1.303	1.684	2.021	2.423	2.704	3.307	3.551
60	0.000	0.679	0.848	1.045	1.296	1.671	2.000	2.390	2.660	3.232	3.460
80	0.000	0.678	0.846	1.043	1.292	1.664	1.990	2.374	2.639	3.195	3.416
100	0.000	0.677	0.845	1.042	1.290	1.660	1.984	2.364	2.626	3.174	3.390
1000	0.000	0.675	0.842	1.037	1.282	1.646	1.962	2.330	2.581	3.098	3.300
z	0.000	0.674	0.842	1.036	1.282	1.645	1.960	2.326	2.576	3.090	3.291
	0%	50%	60%	70%	80%	90%	95%	98%	99%	99.8%	99.9%
	Confidence Level										

Appendix B: Sample of Context Data Collected from Fourth Experiment (IIH)

Log	Device Lo	Availabilit	Posture	Inferred A	I/O Locati	Light	Temperature	Humidity	Proximity	Sound	Orientatic	Network	Battery	Charging
1	In Hand	Free	Upright	null	Indoor[H]	Normal 21.0 lux	null	null	Far 8.0 cm	Silent 39.4 dB	Face Up	Connecte	High 75%	Discharging
2	In Hand	Free	Upright	null	Indoor[H]	Normal 21.0 lux	null	null	Far 8.0 cm	Silent 32.2 dB	Face Up	Connecte	High 75%	Discharging
3	In Hand	Free	Upright	null	Indoor[H]	Normal 21.0 lux	null	null	Far 8.0 cm	Silent 31.9 dB	Face Up	Connecte	High 75%	Discharging
4	In Hand	Free	Upright	null	Indoor[H]	Normal 21.0 lux	Mild 24.450014 Â°C	Medium 52.399574 %	Far 8.0 cm	Silent 31.4 dB	Face Up	Connecte	High 75%	Discharging
5	In Hand	Free	Upright	null	Indoor[H]	Normal 21.0 lux	Mild 24.450014 Â°C	Medium 52.399574 %	Far 8.0 cm	Silent 31.5 dB	Face Up	Connecte	High 75%	Discharging
6	In Hand	Free	Upright	Working	Indoor[H]	Normal 21.0 lux	Mild 24.450014 Â°C	Medium 52.399574 %	Far 8.0 cm	Silent 29.7 dB	Face Up	Connecte	High 75%	Discharging
7	In Hand	Free	Upright	Working	Indoor[H]	Normal 21.0 lux	Mild 24.450014 Â°C	Medium 52.399574 %	Far 8.0 cm	Silent 29.9 dB	Face Up	Connecte	High 75%	Discharging
8	In Hand	Free	Upright	Working	Indoor[H]	Normal 21.0 lux	Mild 24.302956 Â°C	Medium 53.076973 %	Far 8.0 cm	Silent 30 dB	Face Up	Connecte	High 75%	Discharging
9	In Hand	Free	Upright	Working	Indoor[H]	Normal 21.0 lux	Mild 24.302956 Â°C	Medium 53.076973 %	Far 8.0 cm	Silent 31 dB	Face Up	Connecte	High 75%	Discharging
10	In Hand	Free	Upright	Working	Indoor[H]	Normal 22.0 lux	Mild 24.302956 Â°C	Medium 53.076973 %	Far 8.0 cm	Silent 31.3 dB	Face Up	Connecte	High 75%	Discharging
11	In Hand	Free	Upright	Working	Indoor[H]	Normal 21.0 lux	Mild 24.302956 Â°C	Medium 53.076973 %	Far 8.0 cm	Silent 30.4 dB	Face Up	Connecte	High 75%	Discharging
12	In Hand	Free	Upright	Working	Indoor[H]	Normal 21.0 lux	Mild 24.20088 Â°C	Medium 52.27246 %	Far 8.0 cm	Silent 28.7 dB	Face Up	Connecte	High 75%	Discharging
13	In Hand	Free	Upright	Working	Indoor[H]	Normal 21.0 lux	Mild 24.20088 Â°C	Medium 52.27246 %	Far 8.0 cm	Very silent 27.7 dB	Face Up	Connecte	High 75%	Discharging
14	In Hand	Free	Upright	Working	Indoor[H]	Normal 21.0 lux	Mild 24.20088 Â°C	Medium 52.27246 %	Far 8.0 cm	Silent 32.4 dB	Face Up	Connecte	High 75%	Discharging
15	In Hand	Free	Upright	Working	Indoor[H]	Normal 21.0 lux	Mild 24.143864 Â°C	Medium 51.015217 %	Far 8.0 cm	Silent 31.2 dB	Face Up	Connecte	High 75%	Discharging
16	In Hand	Free	Upright	Working	Indoor[H]	Normal 21.0 lux	Mild 24.143864 Â°C	Medium 51.015217 %	Far 8.0 cm	Silent 32.2 dB	Face Up	Connecte	High 75%	Discharging
17	In Hand	Free	Upright	Working	Indoor[H]	Normal 21.0 lux	Mild 24.143864 Â°C	Medium 51.015217 %	Far 8.0 cm	Silent 32.8 dB	Face Up	Connecte	High 75%	Discharging
18	In Hand	Free	Upright	Working	Indoor[H]	Normal 21.0 lux	Mild 24.143864 Â°C	Medium 51.015217 %	Far 8.0 cm	Silent 31.4 dB	Face Up	Connecte	High 75%	Discharging
19	In Hand	Free	Upright	Working	Indoor[H]	Normal 21.0 lux	Mild 24.044235 Â°C	Medium 49.968376 %	Far 8.0 cm	Silent 32.7 dB	Face Up	Connecte	High 75%	Discharging
20	In Hand	Free	Upright	Working	Indoor[H]	Normal 21.0 lux	Mild 24.044235 Â°C	Medium 49.968376 %	Far 8.0 cm	Silent 36.9 dB	Face Up	Connecte	High 75%	Discharging
21	In Hand	Free	Upright	Resting	Indoor[H]	Normal 21.0 lux	Mild 24.044235 Â°C	Medium 49.968376 %	Far 8.0 cm	Silent 38.6 dB	Face Up	Connecte	High 75%	Discharging
22	In Hand	Free	Upright	Resting	Indoor[H]	Normal 21.0 lux	Mild 24.044235 Â°C	Medium 49.968376 %	Far 8.0 cm	Silent 31.4 dB	Face Up	Connecte	High 75%	Discharging
23	In Hand	Free	Upright	Resting	Indoor[H]	Normal 21.0 lux	Mild 24.073217 Â°C	Medium 48.994293 %	Far 8.0 cm	Silent 35.8 dB	Face Up	Connecte	High 75%	Discharging
24	In Hand	Free	Upright	Resting	Indoor[H]	Normal 21.0 lux	Mild 24.073217 Â°C	Medium 48.994293 %	Far 8.0 cm	Silent 33.5 dB	Face Up	Connecte	High 75%	Discharging
25	In Hand	Free	Upright	Resting	Indoor[H]	Normal 21.0 lux	Mild 24.073217 Â°C	Medium 48.994293 %	Far 8.0 cm	Silent 34.3 dB	Face Up	Connecte	High 75%	Discharging
26	In Hand	Free	Upright	Resting	Indoor[H]	Normal 21.0 lux	Mild 24.073217 Â°C	Medium 48.994293 %	Far 8.0 cm	Silent 33.3 dB	Face Up	Connecte	High 75%	Discharging
27	In Hand	Free	Upright	Resting	Indoor[H]	Normal 21.0 lux	Mild 24.140835 Â°C	Medium 48.041428 %	Far 8.0 cm	Silent 37.1 dB	Face Up	Connecte	High 75%	Discharging
28	In Hand	Free	Upright	Resting	Indoor[H]	Normal 21.0 lux	Mild 24.140835 Â°C	Medium 48.041428 %	Far 8.0 cm	Silent 33.8 dB	Face Up	Connecte	High 75%	Discharging
29	In Hand	Free	Upright	Resting	Indoor[H]	Normal 21.0 lux	Mild 24.140835 Â°C	Medium 48.041428 %	Far 8.0 cm	Silent 32 dB	Face Up	Connecte	High 75%	Discharging
30	In Hand	Free	Upright	Resting	Indoor[H]	Normal 21.0 lux	Mild 24.140835 Â°C	Medium 48.041428 %	Far 8.0 cm	Silent 28.6 dB	Face Up	Connecte	High 75%	Discharging
31	In Hand	Free	Upright	Resting	Indoor[H]	Normal 21.0 lux	Mild 23.999842 Â°C	Medium 47.395092 %	Far 8.0 cm	Silent 29.9 dB	Face Up	Connecte	High 75%	Discharging
32	In Hand	Free	Upright	Resting	Indoor[H]	Normal 21.0 lux	Mild 23.999842 Â°C	Medium 47.395092 %	Far 8.0 cm	Silent 28.7 dB	Face Up	Connecte	High 75%	Discharging
33	In Hand	Free	Upright	Resting	Indoor[H]	Normal 21.0 lux	Mild 23.999842 Â°C	Medium 47.395092 %	Far 8.0 cm	Silent 31.4 dB	Face Up	Connecte	High 75%	Discharging
34	In Hand	Free	Upright	Resting	Indoor[H]	Normal 21.0 lux	Mild 23.999842 Â°C	Medium 47.395092 %	Far 8.0 cm	Very silent 27.4 dB	Face Up	Connecte	High 75%	Discharging
35	In Hand	Free	Upright	Resting	Indoor[H]	Normal 21.0 lux	Mild 23.94588 Â°C	Medium 46.798546 %	Far 8.0 cm	Silent 28.5 dB	Face Up	Connecte	High 75%	Discharging
36	In Hand	Free	Upright	Resting	Indoor[H]	Normal 21.0 lux	Mild 23.94588 Â°C	Medium 46.798546 %	Far 8.0 cm	Silent 30.2 dB	Face Up	Connecte	High 75%	Discharging
37	In Hand	Free	Upright	Resting	Indoor[H]	Normal 21.0 lux	Mild 23.94588 Â°C	Medium 46.798546 %	Far 8.0 cm	Silent 29.2 dB	Face Up	Connecte	High 75%	Discharging
38	In Hand	Free	Upright	Resting	Indoor[H]	Normal 21.0 lux	Mild 23.94588 Â°C	Medium 46.798546 %	Far 8.0 cm	Silent 29.1 dB	Face Up	Connecte	High 75%	Discharging
39	In Hand	Free	Upright	Resting	Indoor[H]	Normal 21.0 lux	Mild 23.890684 Â°C	Medium 46.322037 %	Far 8.0 cm	Silent 28.7 dB	Face Up	Connecte	High 75%	Discharging
40	In Hand	Free	Upright	Resting	Indoor[H]	Normal 21.0 lux	Mild 23.890684 Â°C	Medium 46.322037 %	Far 8.0 cm	Silent 28.3 dB	Face Up	Connecte	High 75%	Discharging
41	In Hand	Free	Upright	Resting	Indoor[H]	Normal 21.0 lux	Mild 23.890684 Â°C	Medium 46.322037 %	Far 8.0 cm	Silent 28.4 dB	Face Up	Connecte	High 75%	Discharging
42	In Hand	Free	Upright	Resting	Indoor[H]	Normal 21.0 lux	Mild 23.793808 Â°C	Medium 45.98972 %	Far 8.0 cm	Very silent 27.2 dB	Face Up	Connecte	High 75%	Discharging
43	In Hand	Free	Upright	Resting	Indoor[H]	Normal 21.0 lux	Mild 23.793808 Â°C	Medium 45.98972 %	Far 8.0 cm	Silent 29 dB	Face Up	Connecte	High 75%	Discharging
44	In Hand	Free	Upright	Resting	Indoor[H]	Normal 21.0 lux	Mild 23.793808 Â°C	Medium 45.98972 %	Far 8.0 cm	Silent 28.5 dB	Face Up	Connecte	High 75%	Discharging
45	In Hand	Free	Upright	Resting	Indoor[H]	Normal 21.0 lux	Mild 23.793808 Â°C	Medium 45.98972 %	Far 8.0 cm	Silent 28.6 dB	Face Up	Connecte	High 75%	Discharging
46	In Hand	Free	Upright	Resting	Indoor[H]	Normal 21.0 lux	Mild 23.821136 Â°C	Medium 45.5402 %	Far 8.0 cm	Very silent 27.9 dB	Face Up	Connecte	High 75%	Discharging
47	In Hand	Free	Upright	Resting	Indoor[H]	Normal 21.0 lux	Mild 23.821136 Â°C	Medium 45.5402 %	Far 8.0 cm	Very silent 27.5 dB	Face Up	Connecte	High 75%	Discharging
48	In Hand	Free	Upright	Resting	Indoor[H]	Normal 21.0 lux	Mild 23.821136 Â°C	Medium 45.5402 %	Far 8.0 cm	Very silent 27.4 dB	Face Up	Connecte	High 75%	Discharging
49	In Hand	Free	Upright	Resting	Indoor[H]	Normal 21.0 lux	Mild 23.821136 Â°C	Medium 45.5402 %	Far 8.0 cm	Very silent 26.7 dB	Face Up	Connecte	High 75%	Discharging
50	In Hand	Free	Upright	Resting	Indoor[H]	Normal 21.0 lux	Mild 23.766895 Â°C	Medium 45.356182 %	Far 8.0 cm	Very silent 27 dB	Face Up	Connecte	High 75%	Discharging

Appendix C: Sample of QoC Data Collected from Fourth Experiment (IIH)

2	Light	Normal 21	418	-1	0	0.633333	0.9		1	1	21.40153	20.84847				
2	Temperature															
2	Humidity															
2	Proximity	Far 8.0 cm	1418	417	0	0.633333	1									
2	Sound	Silent 32.2	206	1206	1	0.966667	0.9		1	1	32.4653	31.9747				
2	Orientatic	Face Up -0	264	1264	1	0.966667	0.9			1	0	0				
2	Network	Wifi enab	1420	380	0	0.566667	0.7	0.5								
2	Battery	0.75%	1420	297	0	0.5	1									
2	Charging	Dischargir	1420	300	0	0.666667	1									
2	Ringer	Silent	1421	302	0	0.666667	1									
2	Calendar	None Set	2428	60000	1	0.966667	0.9									
2	Activity															
2	Location	238 Hawo	1426	1000	0	0.426667	0.6	0.3	0.8				1471.0m			
2	Weather															
2	Time of D	Afternoor	1000	1000	1	1	1									
2	Day of the	Tuesday	1000	1000	1	1	1									
2	Device Lo	In Hand	267	1000	1			0.7						0.665	20	19
2	Availabilit	Free	270	998	1			0.7						0.675	28	27
2	Posture	Upright	272	999	1			0.7						0.665	20	19
2	Inferred Activity															
2	I/O Locati	Indoor[H]	277	998	1			0.7						0.646154	13	12
3	Light	Normal 21	705	1705	1	0.966667	0.9		1	1	21	21				
3	Temperature															
3	Humidity															
3	Proximity	Far 8.0 cm	2425	417	0	0.633333	1									
3	Sound	Silent 31.8	416	416	1	0.966667	0.9		1	1	32.32489	31.39511				
3	Orientatic	Face Up -0	223	1222	1	0.966667	0.9			1	0	0				
3	Network	Wifi enab	2427	380	0	0.566667	0.7	0.5								
3	Battery	0.75%	2427	297	0	0.5	1									
3	Charging	Dischargir	2428	300	0	0.666667	1									
3	Ringer	Silent	2428	302	0	0.666667	1									
3	Calendar	None Set	3435	60000	1	0.966667	0.9									
3	Activity															
3	Location	238 Hawo	2429	1000	0	0.426667	0.6	0.3	0.8				1471.0m			
3	Weather															
3	Time of D	Afternoor	1000	1000	1	1	1									
3	Day of the	Tuesday	1000	1000	1	1	1									
3	Device Lo	In Hand	419	1000	1			0.7						0.677419	31	30
3	Availabilit	Free	422	1000	1			0.7						0.682051	39	38
3	Posture	Upright	424	999	1			0.7						0.677419	31	30
3	Inferred Activity															
3	I/O Locati	Indoor[H]	426	999	1			0.7						0.670833	24	23
4	Light	Normal 21	150	2150	1	0.966667	0.9		1	1	21	21				
4	Temperat	Mild 24.45	841	1000	1	0.966667	0.9		1	1	24.57539	24.44855				
4	Humidity	Medium 5	841	1000	1	0.966667	0.9		1	1	52.40976	51.30528				
4	Proximity	Far 8.0 cm	3430	417	0	0.633333	1									
4	Sound	Silent 31.3	224	223	0	0.633333	0.9		1	1	31.70741	31.07259				
4	Orientatic	Face Up -0	961	961	1	0.966667	0.9			1	0	0				
4	Network	Wifi enab	3433	380	0	0.566667	0.7	0.5								
4	Battery	0.75%	3434	297	0	0.5	1									
4	Charging	Dischargir	3434	300	0	0.666667	1									
4	Ringer	Silent	3435	302	0	0.666667	1									
4	Calendar	None Set	4419	60000	1	0.966667	0.9									
4	Activity	Tilting. 10	1419	4000	1	0.9	0.7						100%			
4	Location	238 Hawo	3435	1000	0	0.426667	0.6	0.3	0.8				1471.0m			
4	Weather															
4	Time of D	Afternoor	1000	1000	1	1	1									
4	Day of the	Tuesday	1000	1000	1	1	1									
4	Device Lo	In Hand	225	1000	1			1						0.97561	41	40
4	Availabilit	Free	227	999	1			1						0.979592	49	48
4	Posture	Upright	228	999	1			1						0.97561	41	40