University of
HUDDERSFIELD

Investigation of a Hierarchical Context-aware Architecture for Rule-based Customisation of Mobile Computing Service

Zhaozong Meng

A thesis submitted to the University of Huddersfield
in partial fulfilment of the requirements for
the degree of Doctor of Philosophy

The University of Huddersfield

June 2014

ABSTRACT

The continuous technical progress in mobile device built-in modules and embedded sensing techniques creates opportunities for context-aware mobile applications. The context-aware computing paradigm exploits the relevant context as implicit input to characterise the user and physical environment and provide a computing service customised to the contextual situation. However, heterogeneity in techniques, complexity of contextual situation, and gap between raw sensor data and usable context keep the techniques from truly integration for extensive use. Studies in this area mainly focus on feasibility demonstration of the emerging techniques, and they lack general architecture support and appropriate service customisation strategy.

This investigation aims to provide general system architecture and technical approaches to deal with the heterogeneity problem and efficiently utilise the dynamic context towards proactive computing service that is customised to the contextual situation. The main efforts of this investigation are the approaches to gathering, handling, and utilising the dynamic context information in an efficient way and the decision making and optimisation methods for computing service customisation. In brief, the highlights of this thesis cover the following aspects: (1) a hierarchical context-aware computing architecture supporting interoperable distribution and further use of context; (2) an in-depth analysis and classification of context and the corresponding context acquisition methods; (3) context modelling and context data representation for efficient and interoperable use of context; (4) a rule-based service customisation strategy with a rule generation mechanism to supervise the service customisation.

In addition, feasibility demonstration of the proposed system and contribution justification of this investigation are conducted through case studies and prototype implementations. One case study uses mobile built-in sensing techniques to improve the usability and efficiency of mobile applications constrained by resource limitation, and the other employs the mobile terminal and embedded sensing techniques to predict users' expectations for home facility automatic control. Results demonstrate the feasibility of the proposed context handling architecture and service customisation methods. It shows great potential for employing the context of the computing environment for context-aware adaptation in pervasive and mobile applications but also indicates some underlying problems for further study.

Keywords: Mobile computing, Context awareness, Service customisation, Rule generation, Built-in sensors, Embedded electronics

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

GLOSSARY OF ACRONYMS

The following table gives the acronyms and abbreviations used throughout the thesis.

Acronyms	Full Expression
ADC	Analogue to Digital Converter
ADL	Activity of Daily Living
ANFIS	Adaptive Neuro-Fuzzy Inference System
ANN	Artificial Neural Network
ARM	Advanced RISC Machine
BN	Bayesian Network
CC/PP	Composite Capability/Preference Profile
DIS	Decision Information System
DSP	Digital Signal Processor
DSR	Design Science Research
DT	Decision Tree
ECA	Event-Control-Action
FL	Fuzzy Logic
FMADM	Fuzzy Multi-Attribute Decision Making
GCSM	Generalised Cosine-Similarity Measure
GRBAC	Generalised Role-based Access Control
GRCAC	Grouped Rule-Based Context Access Control
HCAC	Hierarchical Context-Aware Computing
HCI	Human Computer Interaction
HMI	Human Mobile Interaction
HMM	Hidden Markov Model
ICT	Information and Communication Technology
IMEI	International Mobile Equipment Identity
IoT	Internet of Things
JSON	Javascript Object Notation
kNN	K-Nearest Neighbour
LCA	Lowest Common Ancestor
MAUT	Multi-Attribute Utility Theory
MCU	Microcontroller Unit
MES	Mobile Exam System
MSN	Mobile Social Network
MVC	Model View Controller
NFC	Near Field Communication
OWL	Web Ontology Language
QoC	Quality of Context
RBSC	Rule-Based Service Customisation
RESTful	Representational State Transfer
RFID	Radio Frequency Identification
RST	Rough Set Theory
STDEV	Standard Deviation
SVM	Support Vector Machine
UAProf	User Agent Profile
WAP	Wireless Access Point
WSN	Wireless Sensor Network
WSAN	Wireless Sensor Actuator Network
XML	Extensible Markup Language

Chapter 1 Introduction

The technical advances in mobile devices, especially the smartphone built-in and embedded sensing techniques, has motivated people to think about how to effectively utilise the techniques to enhance the usability and efficiency of various computing services. The fusion and penetration of the enabling techniques have promoted the transition of the computing paradigm, from computing with explicit requests to context-aware computing which takes advantage of various sensing techniques to provide a computing service that is customised to the implicit contextual situation (Musumba and Nyongesa, 2013).

Mobile devices such as smartphones, tablets, and some other handheld electronics have evolved from the traditional telecommunication or entertainment electronics to integrated digital terminals for “anytime” and “anywhere” computing service. Compared to the other candidates, smartphones are prominent choices because of the advantages in low cost, small size, wireless connection, data storage, local computation, built-in sensors, and most importantly the widely acceptance amongst users (Jantunen et al., 2008). Therefore, the mobile device is an ideal intermediary between the human and the computing system to obtain users’ requests and collect the context information. These enabling techniques promote prosperity of context-aware mobile applications, such as Location Based Service (LBS), information recommendations, information adaptation, human behaviour recognition, etc. These applications can be employed for extensive use in different areas, such as learning and education, health monitoring, elderly care, intelligent transportation, smart environment, energy saving, and even social activities (Boldrini et al., 2010; Liptchinsky et al., 2014). The computing service is customised to the computational situation by flexible perception of the ambient context with the unobtrusive sensing techniques.

The context-aware computing paradigm is not a new research topic. However, its integration with the smartphone mobile devices using novel sensing techniques to enhance the mobile computing service and facilitate people’s personal and social activities has recently been recognised as a separate area of research. Some research works strive to implement mobile technologies into practical use in different areas, and there is a transition of research from traditional stationary computing systems to mobile systems. The heuristic investigations in this up-and-coming and continuous evolving area create a number of use cases. However, most of the studies focus on the feasibility demonstration of the emerging techniques in identification, communication, and interaction towards context-aware systems (Ding et al., 2011). There are few studies providing holistic solutions for the mobile-based context-aware computing service, as it involves a wide range of techniques and also requires human participation in many application scenarios which may be very difficult to implement and evaluate. Even some critical terms still lack explicit and widely agreed definitions, such as Quality of Context (QoC) (Bellavista et al., 2013).

Although context-aware mobile systems are still at an immature stage, it is a promising way to promote the performance of mobile applications. On one hand, the wide penetration of mobile applications and users' greedy expectations propose requirements for improving user interaction to reduce users' efforts in the applications. On the other hand, cost reduction in the embedded electronics provides chances to use the smartphone and low cost commercially available electronics for context-aware automation to assist people's daily lives. Therefore, it is significant to chart the key issues in this area, and provide solutions to effectively utilise the context and provide an appropriate computing service.

1.1 Motivation

The smartphone today plays a very important role as a digital assistant for a large number of people. Due to the heavy use of smartphones and mobile applications, users demand more efficient and intelligent applications providing better user experience. The context-aware computing paradigm is a promising strategy which customises the computing service to the contextual situation and hence reduces human effort. However, there are many technical barriers in implementing context-aware applications in practical use, and the integration of contextual information into the computation is not the default case of any computing system. Until recently, the main constraints have been the lack of general context-aware mechanisms, heterogeneity of techniques, knowledge-based automation (Carreras et al., 2010), and privacy issues, etc.

People's increasing expectation of mobile applications, constrained by resource limitation and the prosperity of enabling techniques, have motivated the work of this investigation to unobtrusively perceive the contextual situation with the sensing techniques and provide the customised computing service accordingly. It is evident that the context-aware computing concept can be considered to be a promising solution improving the performance of mobile applications by characterising the user and physical environment with the sensing techniques (Anagnostopoulos et al., 2007). Thus, in this investigation, the methods of gathering and utilising the context information, the mechanism for context distribution, and strategy for computing service customisation will be investigated towards a holistic scheme for context-aware mobile computing.

1.2 Focus and Scope

This investigation involves several research fields in the Information and Communication Technology (ICT) area, such as sensor networks, embedded systems, wireless communication, mobile computing, computing services, etc. Therefore, a lot of techniques and methods are required to fulfil the context-aware computing tasks. However, the focus of this research is to effectively utilise the sensing techniques to provide a computing service tailored to the contextual situation. Besides, context acquisition, context modelling, and context-aware service provisioning are integral parts of the research work. Security and privacy of context data are important issues as context data may include users' privacy or confidential data (Lukowicz et al., 2012), but they are beyond the emphasis of this research.

The goal of this investigation is to seek the answers to the research question: "How to integrate the emerging sensing techniques to help the mobile computing system better understand users' requirements and provide proactive computing service that is customised to the contextual situation?" From a technical perspective, this question can be broken down into the following sub-questions:

(1) What are the primary challenges in implementing context-aware computing systems with the latest mobile platforms and novel sensing techniques?
(2) How to observe the context information and integrate the context data into the computation to improve the performance of mobile computing systems?
(3) How to convert the raw sensor data into context values applicable to the applications, and how to model and represent the context data for efficient and interoperable use?
(4) How to effectively utilise the context information and customise the computing service to the contextual situation to better satisfy users' expectation?

This research work strives to seek the answers or solutions for the above questions by using literature surveys, designs and developments, and experimental studies. The purpose of this research is to provide a systematic solution for efficient implementation of the novel built-in and embedded sensing techniques to tailor the computing service to the contextual situation. The proposed methods are implemented with case studies using the low cost commercially available electronics, such as Microcontroller Units (MCU), WiFi transceivers, smartphones, and various sensors.

1.3 Summary of Contributions

This investigation introduces methods for efficient and interoperable discovery and usage of context information with general system architecture for pervasive and mobile computing scenarios. It identifies the critical issues in context-aware mobile systems, and provides methods to take advantage of the context data as implicit input to enhance the user experience. The proof of concept implementation of the proposed system architecture and methods shows great potential. The major contributions of this thesis can be summarised as follows:

(1) A hierarchical system architecture for efficient and interoperable use of context with a rule generation layer to take advantage of history context for further use;

(2) An in-depth analysis and taxonomy of context and sensing techniques in terms of smartphones and widespread low cost commercially available sensory electronics;

(3) Methods for context modelling and context representation considering the heterogeneity of techniques and imperfection of context data;

(4) A computing service customisation strategy based on a rule-based method for decision making and a rule generation method to supervise computing service customisation.

It faces a lot of challenges in putting the proposed solutions into practical application, such as the various techniques involved, difficulty in characterising context situation with sensors, human involvement, etc. However, the case studies produce good results from implementing the methods to promote the performance of mobile applications and for home facility context-aware automation. Thus, this investigation provides referential methods to integrate the context information to promote efficiency and usability of the computing systems.

1.4 Thesis Outline

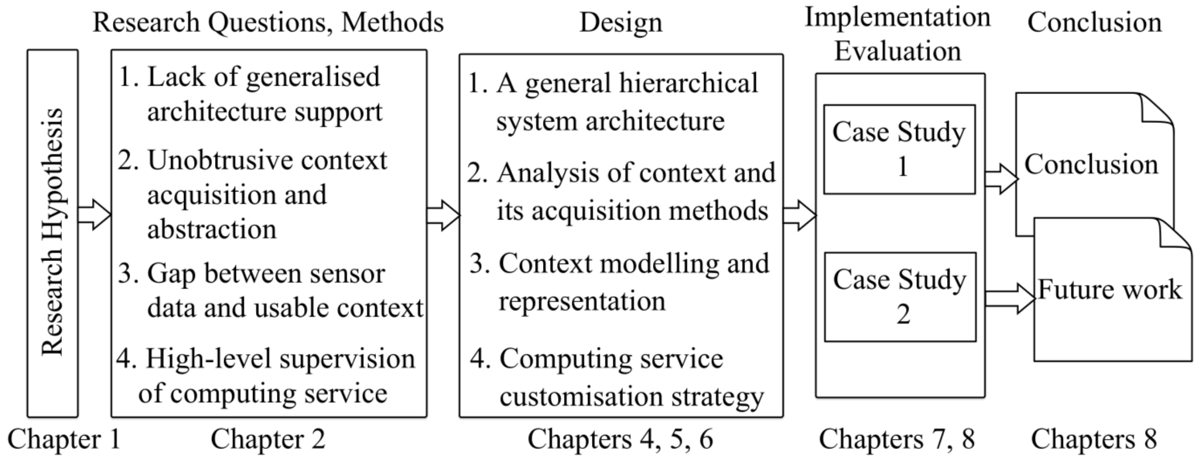

Research Questions, Methods
Design
Implementation Evaluation
Conclusion
Research Hypothesis
1. Lack of generalised architecture support
2. Unobtrusive context acquisition and abstraction
3. Gap between sensor data and usable context
4. High-level supervision of computing service
1. A general hierarchical system architecture
2. Analysis of context and its acquisition methods
3. Context modelling and representation
4. Computing service customisation strategy
Case Study 1
Case Study 2
Conclusion
Future work
Chapter 1
Chapter 2
Chapters 4, 5, 6
Chapters 7, 8
Chapters 8

Figure 1-1 Thesis Structure

The following chapters of this thesis put the research hypothesis into practice through system architecture design, methods and models, and case studies and prototype implementations. As is shown in Figure 1-1, the remainder of this thesis is organised as follows:

Chapter 2 presents an overview on the conceptual framework of context-aware mobile computing systems. The review work highlights the understanding of context and context-awareness, and then provides insight into the enabling techniques, successful applications, context-aware computing frameworks, and context-aware service adaptation models. Then, the challenges facing the efficient use of context for service customisation are determined.

Chapter 3 gives the aim and objectives of this investigation with the corresponding research methods to achieve the research purpose.

Chapter 4 presents a hierarchical context-aware computing architecture for context gathering, handling, distribution, and context-aware service provisioning. In order to deal with the heterogeneous techniques and information of different semantic level, the architecture is divided into four hierarchical layers: physical layer, context handling layer, context-aware service layer, and rule generation layer. The functional modules are explicitly defined and the relevant techniques and methods are illustrated to enhance the efficiency and interoperability.

Chapter 5 introduces the context classification methods based on in-depth understanding of prospective context information and built-in and embedded sensing techniques. The relevant context acquisition, context modelling, and data representation methods are provided for efficient and interoperable use of context.

Chapter 6 focuses on context-aware service provisioning based on the proposed computing service customisation strategy. A rule-based method is employed for decision making about service customisation and a rule generation method is employed to generate new rules for high-level supervision. The performances of both methods are evaluated through simulations and then practised in the case studies.

Chapter 7 discusses evaluation of context-aware mobile system with respect to its features.

Chapter 8 provides two case studies under the framework and methods described in Chapters 4, 5, and 6. Case study 1 is on improvement of usability and efficiency of mobile applications with mobile built-in sensing techniques, and case study 2 is on smart home facility control with mobile terminals and embedded sensing techniques. Experimental studies demonstrate the feasibility of the proposed solution but also reveal the inadequacies for further study.

Finally, conclusions are drawn and future work is suggested in Chapter 9.

Chapter 2 Literature Review

The field of context-aware mobile computing is increasingly gaining applicability with the progress of enabling techniques. In order to effectively utilise the techniques to provide users with appropriate computing services, this chapter will provide some insight into context-aware mobile systems and identify the challenges facing this research area. In the following, the concepts of context and context awareness and the features of context data are introduced, the enabling techniques and various context-aware applications are discussed, and the major concerns in context-aware mobile computing, such as architecture support and computing service customisation methods, are summarised and analysed.

2.1 Background

In traditional computing systems, users make requests of the computing service via explicit inputs with the mouse and keyboard. But for some resource limited devices such as smartphones, the efficiency and arbitrary degree of content in user interaction is undoubtedly restricted by the screen size (Dey, 2000). However, the computing service enabled by the easily accessible and low cost wireless infrastructure has been set free from location constraints. Thus, mobile devices such as smartphones and tablets are widely accepted for ‘anytime’ and ‘anywhere’ computing services. The permeation of the computing service into people’s daily life promotes users’ expectation in efficiency and conveniency in user interaction.

However, computing that can facilitate people’s daily life is not limited to smartphones. The various sensing techniques and embedded electronics have promoted the evolvement of computing systems being aware of the ambient environment. These sensing techniques within the coverage of wireless infrastructure can reduce human efforts in applications or intelligently execute computing services to promote user experience. Moreover, they can even provide some assistance that was previously impossible without the relevant techniques. The Wireless Sensor Actuator Network (WSAN), a new generation of sensor networks, has become an active research area in which sensors and actuators can interact with the physical world to serve people (Xia et al., 2007; Zeng et al., 2013). It is estimated that there will be over 50 billion devices connected to the Internet by 2020, and they are now promoting a new paradigm in which every object becomes interactive (Feki et al., 2013; Lumpkin, 2013).

It is self-evident that efficient interaction between user and computing system is an essential requirement to effectively fulfil computing tasks. Context awareness is a promising solution to promote computational intelligence by perceiving and integrating computational contexts such as computation resource, ambient environment, and user preference into the computation to customise the computing service. Users' requests can be extended and enriched with the sensor data as implicit input to refine and customise the computing service. Context awareness was originally the central idea of ubiquitous and pervasive computing, for which context data is collected for service adaptation (Achilleos et al., 2010). The extensive concern with context awareness in the academic area is accompanied by the wide spread of mobile applications. Early studies of context information largely focus on location-based service structures and applications (Yue et al., 2005). The context-aware model for mobile computing proposed by Schilit (1995) in his PhD thesis is considered as the milestone of the early stage.

With the wide permeation of mobile devices such as smartphones and tablets, handheld and wireless connected devices become an ideal medium which can be used to characterise users' situations. On the other hand, the fast evolution of mobile device computation power and wireless networks provides prospective infrastructure gathering the physical context with various built-in and embedded sensing techniques. These techniques have created opportunities for the context-aware computing service, which mainly focuses on the following fields:

- *Location-based service - Navigation and tour guiding*
- *Information recommendation – News or movie recommendation*
- *Human behaviour recognition - Smart user interaction*
- *Smart environment - Smart home and smart office*
- *Elderly care - Emergency alarm, daily life assistant*
- *Healthcare - Health monitoring, automatic diagnosis*
- *Industrial process control (Korber et al., 2007)*

Although there are remarkable advances in the relevant techniques, context awareness in mobile and pervasive systems is still at an immature stage. Relevant studies continuously appear in large numbers, but they lack general architecture and a context-aware computing service strategy to deal with the heterogeneous techniques and provide an appropriate computing service. Most of the research works are application-specific implementations in particular physical areas or with limited quantity of context parameters which are lack of generality (Bellavista et al., 2013). Therefore, it is significant to chart the focal issues in this research area, and propose a general architecture and design methodology with holistic technical solutions to integrate the context into computation for an appropriate computing service. The following sections provide insight into the concept, enabling techniques, related work, and some critical issues for this particular area.

2.2 Understanding Context and Context Awareness

The concept of context-aware computing can be tracked down to the days when Mark Weiser published the paper "The Computer for the 21st Century" (Weiser, 1991). Weiser envisaged future computing systems knowing their location, being able to obtain information and receive context information, and offering seamless interaction to assist users' tasks in a transparent way. The terms context and context awareness now have a wide range of meanings in different use scenarios, and several definitions can be found in recent publications. This section will give a clear view of the terminologies to better point out the scope of this research and facilitate the development of context-aware computing systems.

2.2.1 Defining Context and Context Awareness

In the study that first introduces the term context awareness, Schilit and Theimer (1994) refer to context as location, identities of nearby people and objects, and changes to those objects. Schilit (1994) characterises context as a collection of information that describes the users in a context-aware system; it is defined as: "*In a mobile distributed computing system, contexts are the location of the user, the identity of people and physical objects that are nearby the user and the states of devices that the user interact with*". Currently, the widely accepted definition for context is by Dey (2000): "*Context is any information that can be used to characterise 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 application themselves*". In this definition, context means all about the whole situation relevant to an application and its set of users (Dey and Abowd, 1999). Bolchini et al. (2009) define context "*the set of variables that may be of interest for an agent and that influence its actions*".

Then, in the work by Chen and Kotz (2000), the definition covers the main context aspects with a straightforward classification, which defines context as "*a four-dimensional space composed by: computing context, physical context, time context, and user context.*" Similarly, Zimmermann et al. (2007) argue that context is "*elements for the description of this context information fall into five categories: individually, activity, location, time, and relations*".

All the above definitions introduce considerable amount of expert knowledge. Dey's and Bolchini's definitions are generic and abstract to cover most scenarios. Chen and Kotz, and Zimmermann provide explicit classifications of context, which are convenient for practical use. In this investigation, Dey's definition is recognised as the appropriate description of the term context, and the definitions by classification are considered in the context modelling.

When it was first introduced by Schilit and Theimer, context-awareness was regarded as a new class of applications that were aware of the context in which they were executed (Hristova. 2008). Schilit et al. (1994) deem context awareness as the ability to "*adapt according to the location of use, the collection of nearby people, host, accessible devices, as well as to change such things over time*". The definitions of context awareness are largely similar although they may be defined from different perspectives and with particular emphasis. From the service adaptation perspective, Dey and Abowd (1999) define it as: *"A system is context-aware if it uses context to provide relevant information and/or services to the user, where relevancy depends on the user's task"*. Then, from a human computer interaction perspective, it is defined by Daniele (2006): *"Context-aware applications can determine their behaviour by sensing and exploring the user's content without explicit user intervention. They can intelligently react upon changes in the user's context performing actions relevant to the user, the application itself, and the interaction between user and application"*. Since the purpose of this research is to customise the computing service with context information, Dey and Abowd's definition is the most suitable one for this work.

From the above discussion, context awareness is the capability of computing systems to provide the user with a computing service dynamically customised to the context, which is collected as an implicit input into the system. Context awareness is realised mainly by acquisition of context information and context-aware service adaptation. It is an effective way to reduce human effort and improve the usability and efficiency of the computing system.

2.2.2 Imperfection Features of Context and QoC

As context involves real world entities, it is therefore subject to uncertainty and inaccuracy by its nature (Preuveneers and Berbers, 2006). Many studies have been aware of the uncertainty, noise, and error-prone problems as general features of smartphone and embedded sensor context data. These features are characterised by some properties referred to as QoC indicators in literature (Badidi and Esmahi, 2011). It is undoubted that the accuracy of the context data may largely affect performance of the context-aware computing service. Thus, it is an essential task to incorporate the QoC parameters into the computation in the process of context acquisition, distribution, and context-aware service customisation. In some studies, QoC modelling is regarded as a prerequisite that should be considered before creating service infrastructure for context-aware service provisioning (Salden et al., 2004).

On the other hand, scholars argue that the set of QoC is directly application-independent because the relevance of QoC indicators changes in accordance with information type of context considered and application/user requirements (Filho and Martin, 2008). With an overview of QoC in the existing investigations, the challenges facing the application of QoC in context-aware service customisation may fall into the following aspects:

- Definition of QoC in context-aware computing systems,
- Determination of the parameters describing the QoC,
- Normalization and representation of QoC for interoperable use, and
- QoC mechanism for efficient context-aware service customisation.

Although the quality of the context information has already attracted much attention for its importance in context-aware service customisation, it is still an emerging technical term without a certain definition. Buchholz et al. (2003) have defined the QoC as: *any information that describes the quality of information that is used as context information*. This definition stresses that QoC refers to information but not the process or the hardware component that could possibly provide the information. Buchholz et al. have also discussed the relationship between QoC, QoD (Quality of Device), and QoS. Hristova (2008) argued that the researchers have distinguished and defined several QoC parameters that better define and describe the concept of QoC: precision, accuracy or correctness, freshness, reliability, and resolution, which can be used to characterise most imperfection features of the context data. There are other attributes of information quality identified in some research work, such as coverage, resolution, accuracy, repeatability, frequency, and timeliness (Gray and Salber, 2001). A new definition for QoC by Bellavista et al. (2013) is: *the set of parameters that express quality requirements and properties for context data (e.g., precision, freshness, trustworthiness, etc.)*. This definition is general and simple, and it is suitable for smartphone based context-aware systems. Manzoor (2010) argued that QoC must also consider the requirements of the context consumer and the intended use of context information, and defined QoC as: "*Quality of context indicates the degree of conformity of the context collected by sensors to the prevailing situation in the environment and the requirements of a particular context consumer*". This definition explains that the context information should be suitable to be used by a specific context consumer for an intended purpose, namely, the context consumer and context have their specificity.

Table 2-1 QoC Parameters

Peer Investigations	**QoC Parameters**												
	Precision	Correctness (Accuracy)	Trustworthiness	Resolution	Up-To-Dateness (Timeliness, Freshness)	Reliability	Validity	Completeness	Significance	Coverage	Repeatability	Frequency	Security
(Buchholz et al., 2003)	O	O	X	O	O	O	X	X	X	X	X	X	X
(Gray and Salber, 2001)	X	O	X	O	O	X	X	X	X	O	O	O	X
(Hristova, 2008)	O	O	O	X	O	O	X	X	X	X	X	X	X
(Abid et al., 2009)	O	O	O	X	O	X	X	O	X	X	X	X	O
(Manzoor et al., 2008)	X	X	O	X	O	X	X	O	O	X	X	X	X
(Bellavista et al., 2013)	O	X	X	X	O	X	O	X	X	X	X	X	X
(Filho et al., 2010)	O	O	X	O	O	X	X	O	X	X	X	X	O
(Sheikh et al., 2008)	O	O	X	O	O	X	X	X	X	X	X	X	X
(Wieland et al., 2009)	X	O	O	O	O	O	X	X	X	X	X	X	X
(Kim and Lee, 2006)	O	O	O	O	O	X	X	X	X	X	X	X	X
(TalebiFard and Leung, 2011)	O	O	O	O	O	X	X	O	X	X	X	X	X

(Symbols: "O" denotes inclusion of the parameter and "X" stands for non-inclusion)

From Table 2-1, it is evident that up-to-dateness is accepted by all the listed investigations. It is most likely because the real-time performance is an essential requirement of the context-aware systems. Besides, the correctness, precision, trustworthiness, and resolution are the parameters also widely accepted to describe the QoC. The other parameters, such as reliability, validity, completeness, coverage, repeatability, frequency, and security may also be important in some use scenarios, though they gained less attention compared with the previous ones. QoC descriptions are undoubtedly useful and necessary, but some of the parameters are very difficult to acquire. The over use of irrelevant parameters may be a risk in overloading the lightweight computing system of the sensing nodes and mobile phone.

2.2.3 Context Modelling

In order to customise the computing service to the contextual situation, the context data describing the contextual situation should be understood by the computing system. However, much of the context data involved is obtained via sensing techniques, and there is usually a significant gap between the raw sensor data and the level of information that is useful to the applications (Hoareau and Satoh, 2009). The context model is an important component to formalise the data for further reasoning and processing.

(1) Context model in context-aware systems

Due to the inherent complexity of context-aware systems, development of context-aware applications should be supported by adequate context modelling and reasoning techniques to reduce the complexity and improve the maintainability. The context model provides an unambiguous definition of context information, their representation, semantic and usage which are indispensable to the system. It takes into account the general characteristics of context information, such as its temporal nature, ambiguity, impreciseness, incompleteness and privacy (Reichle et al., 2008). Moreover, a formal representation of context data with a model is necessary for consistency checking, as well as to ensure that sound reasoning can be performed on context data (Bettini et al., 2010). The existing context modelling approaches differ mainly in the aspects of: (1) the convenience with which real world concepts can be captured by computer systems, (2) the expressive power of the context models to represent context information, (3) the support they can provide for context reasoning and processing.

(2) Context modelling approaches

Early approaches for context modelling include key-value model, mark-up-based model, and graphical model (Hoareau and Satoh, 2009). These methods are suitable to deal with context data that is simple and explicit, but not appropriate for the context data with some uncertainty, inaccuracy, and incompleteness. In addition, more complicated information modelling techniques are employed to express context information for context-aware computing. The commonly used context modelling approaches are object-oriented model, logic-based model, and ontology-based model (Bettini et al., 2010).

In general, there is no universal set of context profiles that is valid for all application domains, as context is always an issue of the interaction between a user and an application (Dey, 2001). Therefore, due to the different requirements and features of context, different context modelling approaches may be suitable for the use cases. In addition to the above context modelling approaches, there are hybrid context models which integrate two or more context modelling methods to better represent the context data. The CARE (Agostini et al., 2009) framework adopts a context modelling approach that is based on a loose integration of a mark-up model and an ontological model. Henricksen et al. (2004) proposed a hybrid approach for context modelling combining ontologies with the fact-based approach provided by context modelling language. This method can combine the advantages of the context modelling language model with interoperability support and strong reasoning capability provided by ontological models.

In the survey by Bettini et al. (2010), it is concluded that combining different techniques towards a hybrid approach is promising to fulfil the requirements for context modelling.

(3) Evaluation of context modelling approaches

For the evaluation of context models, Hoareau and Satoh (2009) assorted the methods into two evaluation frameworks: requirement-based evaluation framework and data-based evaluation framework. With the first evaluation framework, Strang and Linnhoff-Popien (2004) evaluated the above modelling approaches based on six requirements a context-aware computing system should satisfy, which are widely accepted. The requirements are:

- *Distributed composition (dc)*
- *Partial validation (pv)*
- *Richness and quality of information (qua)*
- *Incompleteness and ambiguity (inc)*
- *Level of formalism (for)*
- *Application to existing environments (app)*

The above requirements are importance indicators for the evaluation of context modelling approaches dealing with pervasive systems. The result of the evaluation by Strang and Linnhoff-Popien is given in Table 2-2.

Table 2-2 Evaluation of Context Modelling Approaches

Attributes / Modelling Approaches	*dc*	*pv*	*qua*	*inc*	*for*	*app*
Key-value models	/	/	-	-	-	+
Mark-up scheme models	+	++	/	/	+	++
Graphical models	-	/	+	/	+	+
Object-oriented models	++	+	+	+	+	+
Logic-based models	++	/	/	/	++	-
Ontology-based models	++	++	+	+	++	+

(Symbols: “/” denotes not available, “-” for poor, “+” for medium, and “++” for good)

From Table 2-2, we may find ontology-based context modelling makes good performance in *dc*, *pv*, and *for*, and it is not poor in *qua*, *inc*, and *app*. It is widely accepted that ontology-based method makes the best description of context because it allows good sharing of information with common semantics (Sahafipour and Javidan, 2012; Bae, 2014).

2.3 Enabling Techniques

As context-aware systems usually involve a wide range of supporting technologies, this section will give an insight into the present situation of enabling techniques. The context-aware system is motivated by the technical progress in several related fields. However, the most closely related fields may be smartphone technology and sensor technology. The former is responsible for user situation characterisation and user interaction, and the latter is responsible for the various context data acquisition.

2.3.1 Continuous Progress of Mobile Device Capability

The smartphone device is a fast evolving area in the recent years, and a lot of new models have emerged with novel functions. Technical progress happens in nearly every aspect of the mobile devices. These techniques have enhanced the computation and communication power of the devices and allowed them the capability of perceiving the ambient environment. Of course, mobile software platforms have also promoted mobile devices to be employed in different fields and widely owned (Gavalas and Economou, 2011).

Table 2-3 and Table 2-4 list the models of Apple iOS devices and Google Android devices.

Table 2-3 Progress of the Apple iPhone Devices

Model	CPU/Chipset	RAM	Storage(GB)	Display	Release
iPhone (Apple Inc., 2010)	412M ARM 11	128M	4/8/16	3.5 Inches 320*480	2007
iPhone 3G (Apple Inc., 2012a)	412M ARM 11	128M	8/16	3.5 Inches 320*480	2008
iPhone 3GS (Apple Inc., 2012b)	600M ARM Cortex-A8	256M	16/32	3.5 Inches 320*480	2009
iPhone 4 (Apple Inc., 2012c)	600M ARM Cortex-A8/ Apple A4	512M	16/32	3.5 Inches 640*960	2010
iPhone 4S (Apple Inc., 2013a)	800M Dual-core ARM Cortex A9/ Apple A5	512M DDR2	16/32/64	3.5 Inches 640*960	2011
iPhone 5 (Apple Inc., 2013b)	1.3G Dual-core /Apple A6 chipset	1G LPDDR2-1066	16/32/64	4.0 Inches 640*1136	2012
iPhone 5s (Apple Inc., 2013c)	1.3G Dural-core/ Apple A7 chipset with M7 motion coprocessor	1G DDR3	16/32/64	4.0 Inches 640*1136	2013

Table 2-4 Progress of the Google Android Devices

Model	CPU/Chipset	RAM	Storage(GB)	Display	Release
Nexus One (GSMArena, 2013a)	1 GHz Scorpion/ Qualcomm QSD8250 Snapdragon	512M	4	3.7 Inches 480*800	2010
Nexus S (GSMArena, 2013b)	1G Cortex A8/ Samsung Exynos 3	512M	16	4.0 Inches 480*800	2010
Nexus 4 (GSMArena, 2013c)	1.5 GHz quad-core Krait/ Qualcomm APQ8064 Snapdragon	2G	8/16	4.7 Inches 768*1280	2012
Nexus 7 (Tablet) (GSMArena, 2013d)	NVIDIA ® Tegra ® 3 quad-core processor	1G	16/32	7 Inches 800*1280	2012
Nexus 10 (Tablet) (GSMArena, 2013e)	Dual-core A15	2G	16/32	10 Inches 1600*2560	2012
Nexus 5 (GSMArena, 2013f)	Qualcomm Snapdragon^{TM}800, 2.26GHz	2G	16/32	4.95 Inches 1080*1920	2013

From the above tables, the CPU/Chipset, RAM, Storage, and display of the devices have all progressed continuously in recent years. Therefore, the computation power, information presentation, and HCI capability of the mobile devices are promoted. These technical progresses have laid the foundation for context-aware mobile systems.

The primary task of the context-aware system is to gather the context information, distribute the context efficiently, and generate an appropriate computing service. Thus, the most remarkable developments of the devices in supporting context-aware applications are: (1) computation power, (2) wireless connection, and (3) various built-in sensing modules. In addition to technical progress, the diversity and heterogeneity are significant features of the mobile devices.

2.3.2 Mobile Built-in and Embedded Sensing Techniques

Only with the various sensing techniques can the computing system be allowed to gather information of interest to describe the computational context and customise the computing service to the contextual situation. The main sensing techniques available are the mobile device built-in sensors and embedded sensors.

(1) Built-in sensors of the rich sensing mobile devices

A significant success of the smartphone devices which distinguishes the smartphone from the traditional 2nd generation GSM phone are the built-in sensing techniques. Since the mobile devices are personal and handheld, they are the appropriate approach to collect useful information to characterise users (Raento et al., 2005). Table 2-5 gives some mobile built-in sensors that are commonly shipped on smartphone devices.

Table 2-5 Mobile Built-in Sensing Techniques (Apple Inc., 2013c; GSMArena, 2013f)

Built-in Sensing Modules	Potential Use
Touch screen	User interaction
Camera	Object recognition
Microphone	Speech recognition
GPS module	Positioning, navigation
Proximity sensor	Energy saving
Digital Compass	Navigation
Ambient brightness sensor	LCD brightness adjustment
Near Field Communication (NFC) Module	User Identification, mobile payment
3-Axis Accelerometer	Behaviour recognition
Gyroscope sensor	Orientation based control

With the built-in sensing modules given in the table, some useful context information such as location, preference, traveling mode, and ambient environment can be collected to characterise the contextual situation towards adapted computing service. Some mobile sensing frameworks collecting the built-in sensor values with the mobile platforms have emerged, such as ContextPhone (Raento et al., 2005), LifeMap (Chon and Cha, 2011), and G-Sense (Perez et al., 2010).

(2) Embedded sensing techniques

Table 2-6 Example Embedded Sensors

Sensors	Resolution	Power Supply	Digital Output
Brightness (Rohm Co., 2010)	Adjustable	5V	I2C Serial Bus
Obstacle (ElecFreaks, 2013)	10cm-60cm	5V	GPIO
Human (DrRobot Inc., 2005)	NA	5V	GPIO
Humidity (Elecrow, 2013)	Adjustable	5V	GPIO
Smoke (Pololu, 2013)	Adjustable	3-5V	GPIO
Sound (Emartee, 2013)	Adjustable	3-5V	GPIO
Barometer (BOSCH, 2008)	0.06hPa	1.6-3.6V	I2C Serial Bus
Temperature (ElecFreaks, 2011)	0.5℃	3-5.5V	1-Wire Data Bus
Ultrasonic distance (Parallax Inc., 2006)	NA	5V	GPIO

The technical progress in sensor technologies results in cost reduction, miniaturisation and wide application of the independent sensors (Jantunen et al., 2008). The independent sensors can be connected with the embedded electronics to be integrated with the computing systems. Therefore, the environment can be monitored with the sensors flexibly, and automatic control or data analysis can be easily conducted. There are a lot of independent sensors available to perceive and characterise the human and ambient environment to fulfil context-aware computing tasks. Some example embedded sensors and their attributes are given in Table 2-6.

From the table, the embedded sensors are different in power supply, digital interface, and resolution. Therefore, different communication interfaces, data formats, and control logics are required in handling the various sensors and obtaining the useful values. However, the diversity of the sensing techniques and heterogeneity in techniques also raise challenges in their practical application (Markris et al., 2013). It is an important task to deal with the heterogeneity problem and integrate the diverse techniques in an interoperable framework.

Due to the heterogeneity in devices and sensing techniques, it is of great importance to build the system to be compatible with open standards and techniques. Researchers argue that it is important to conform to open standards to deal with the diverse devices for context-aware computing service (Wu et al. 2007; Lin et al., 2008). As a context-aware system involves techniques from the sensor acquisition and devices communications to service discovery and provisioning, to model explicitly the functional modules in a hierarchical system architecture and guarantee the interoperable data interaction is a promising design strategy.

2.4 Applications, Methods, and Mathematical Models

The above sections provide an insight into context-aware mobile computing through the discussion on concept definition, context modelling, and enabling techniques. This section reviews the existing context-aware mobile applications and gives a classification of the existing studies. Up-to-date investigations on the system architectures, design methods, and theories and models in the area of context-aware mobile systems are then provided.

2.4.1 Context-aware Mobile Applications - Taxonomy and Discussion

The context-aware computing paradigm makes mobile computing systems more intelligent by being aware of the context of computing tasks. The mobile terminals which can be used to characterise the users’ preference can facilitate the computing tasks by context-aware adaptation. Thus, cost reduction of mobile devices and various embedded sensing techniques has resulted in a lot of heuristic studies for a large spectrum of use cases. The early applications are mainly location-based systems and some experimental systems with a limited number of context variables. The concept of context awareness then propagates to wide areas of applications, such as mobile browsing, mobile recommendation, Mobile Social Networks (MSN), Human Mobile Interaction (HMI), human behaviour recognition, smart environment, mobile healthcare, and mobile learning, etc. In these studies, the computation power, network condition, and ambient environment are employed as the computational context to improve the performance of the applications. Generally, context-aware systems can be classified into two kinds from the perspective of the purpose using mobile devices: context-aware information retrieval, and context-aware automation.

(1) Context-aware information retrieval

Mobile devices are used as medium for timely and relevant information retrieval or recommendation. The contextual situation is gathered and characterised with various sensing techniques to tailor or refine the content of the retrieved information to fit the contextual situation and satisfy users' preferences.

(2) Context-aware automation and decision support

This kind of context-aware system uses sensor data as context to predict users' expectation and fulfil the computing tasks in advance automatically. It usually involves automatic execution of computation tasks to reduce users' efforts compared with traditional applications.

There are also systems combining (1) and (2) which fall into the overlap area of these two kinds. The relevant studies can be divided into the two kinds as shown in Figure 2-1.

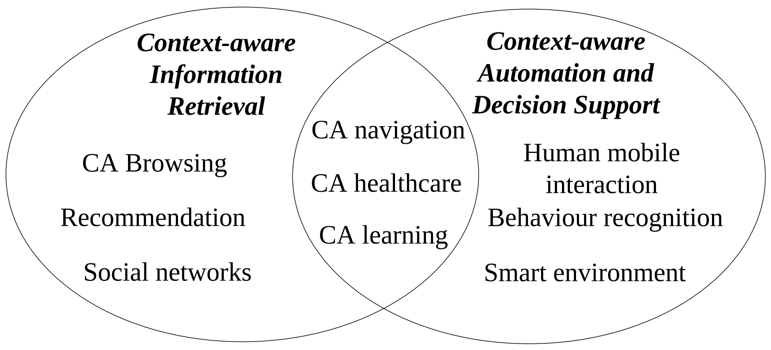


Figure 2-1 Taxonomy of Context-aware Applications

It is not difficult to find that the context-aware system based on the mobile devices is gaining more and more research interests in different areas. New techniques and applications, and novel methods and ideas emerge continuously. In order to provide an insight into the current situation of this research area and identify the difficulties and challenges, the existing works are introduced according to the above classification. Some typical and relevant investigations in different application areas are summarised in Table 2-7.

Table 2-7 Context-aware Mobile Applications- Taxonomy and Examples

Application Areas	Typical Applications
Context-aware browsing	Context-Aware Browser (Coppola et al., 2010; Espada et al., 2012) Web-centric approach (Gossweiler et al., 2011) Context-aware mobile web browsing (Notles, 2008) SENSE-SATION (Shirazi et al., 2010) Context-aware mobile Web2.0 (Koskela et al., 2007; Hsu, 2011) Context-aware mobile browsing based on HTML 5 (Zhang et al., 2012)
Recommendation	Context-aware media recommendation (Yu et al., 2006) Mobile web news recommendation system (Lee and Park, 2007) Product recommendation (Ricco and Bguyen, 2007) Recommendation in mobile commerce (Hosseini-Pozveh, 2009) Easylife restaurant recommendation (Zhuang et al., 2011) Intelligent Web recommendation (Fong et al., 2011)
Social networks	Mobile social health (Baumer et al., 2013) Social collaboration (Liptchinsky et al., 2014) Middleware for mobile social networks (Arnaboldi et al., 2013)
Human mobile interaction	Camera-phones (Toye et al., 2004) NeuroPhone (Campbell et al., 2012) EyePhone (Miluzzo et al., 2010b) SoundSense (Lu et al., 2009)
Behaviour recognition	Fall detection (Sixsmith and Johnson, 2004; Chen et al., 2005; Hansen et al., 2005) Walking detection (Pappas et al., 2004; Barralon, 2006) Human activity recognition with wearable sensors (Huynh, 2008) Automatic activity recommendation system (Choudhury et al., 2008) Human activities with wrist-worn sensor platform (Ward et al., 2006) Continuous human actions classification (Yang et al., 2009) Pervasive fall detection system - PerFallD (Dai et al., 2010)
Smart environment	WSAN (Xia et al., 2007; Chung and Oh, 2006; Liang et al., 2008) Smart home architecture (Chen and Zhi, 2011; Cheng et al., 2012) Smart home elderly care (Medjahed et al., 2011; Charlon et al., 2013) Smart home Activity of Daily Living (ADL) assistant (Bae, 2014) Product automation in factory (Korber et al., 2007) Context-aware meeting room (Grønli et al., 2010)
Navigation	Mobile touring and navigation (Saeedi et al., 2010) Proactive mobile museum guide (Lanir et al., 2011)
Healthcare	Remote healthcare service (Jiang et al., 2008) Patient monitoring and medical diagnosis (Wu et al., 2008) Chronic illness self-management system (Sha et al., 2008) BALANCE (Denning et al., 2009) Hyperfit system (Jarvinen et al., 2008) HealthAware system (Gao et al., 2009)
Learning	Learning content context-aware distribution (Basaeed et al., 2007) Context-aware synchronous learning systems (Huang et al., 2008) Wireless Response System (Lu et al., 2010) Intelligent ambient learning environment (Scott and Benlamri, 2010) Mobile learning adaptation framework (Al-Hmouz et al., 2012) Context-aware mobile learning system (Kasaki et al., 2012) Mobile learning adaptation system (Chrofi, 2012)

(1) Context-aware browsing

Mobile Internet browsing now plays an import role since it is not restricted to fixed location, unlike traditional computing systems. However, it is expected to be more efficient and flexible to access timely and relevant information. The investigations in this area mainly focus on context-aware browser architecture, integration with new techniques and design concepts, and context sensing methods for mobile browsing.

There are investigations striving to improve the system structure of browsers shipped on the mobile devices to integrate the context information in information retrieval. Coppola et al. (2010) proposed a context-aware browser to handle context with a two-stage inferential mechanism combining rules and Bayesian Network (BN). The weakness of this system lies in the limited models, which rely on rigid categorisation built on ontologies and strict terminologies. Espada et al. (2012) proposed a system consisting of a modular Web browser and a set of specific Extensible Markup Language (XML) tags that can be used in Web applications. It contains a context-aware Web browser architecture to exploit device sensors to capture and use context information.

With the progress of mobile platforms, there arise many mobile oriented Web-centric development kits and new techniques, such as PhoneGap, WebOS, and Argos. Gossweiler et al. (2011) introduced a Web-centric approach based on Argos to explore the Android system's rich features to fulfil context-aware service. Users can deploy the Argos's JavaScript library to gain access to native phone resources, such as microphone voice recognition, haptic feedback, Bluetooth, NFC, and graphics, etc. Koskela et al. (2007) identified the potentiality of combining Web techniques and user context in enriching and personalising the computing service with a mobile device, and proposed a context-aware mobile Web service architecture. Hus (2011) focused on the interoperability and reusability among heterogeneous context-aware systems and various mobile devices and proposed a Multi-layer Context Framework that integrates Web technologies into a context-aware system. The mobile Web technologies are adopted as the backbone of context-aware systems to facilitate the sharing and exchange of context-aware resources. A context-aware attendance monitoring system is implemented with ZigBee to demonstrate the feasibility of the infrastructure. Zhang et al. (2012) presented a context-aware mobile browsing system that could reduce useless information and enhance user experience. This system captures user context through mobile built-in sensors, recommends interested information, and then adapts the Web page according to user profile and current context.

Another direction of context-aware mobile browsing is the context sensing capability of mobile devices. In order to make the context data available to Web developers, a sensing platform for mobile devices called SENSE-SATION was proposed by Shirazi et al. (2010). This platform gathers and stores context information on mobile phone and makes it directly accessible via the Representational State Transfer (RESTful) web service. The platform supports the full development process using a concept of virtual sensors. The Web developers are allowed to create applications making use of mobile sensing techniques without mobile phone programming at all. Notles (2008) investigated the context information that could be gathered by mobile phones and proposed an architecture for sending context information from mobile web browser to a web server via HTTP requests for context-aware mobile Web browsing. The architecture consists of a plug-in for a mobile web browser, which gathers context information from the mobile operating system and then sends the data to the server.

As ‘anytime’ and ‘anywhere’ information access becomes more and more prevalent in people’s daily life, context-aware mobile browsing which can improve user interaction and refine the content of information retrieval becomes more significant. However, most of the above mentioned investigations focus on the proposition of new architectures and design concepts. The subject lacks mature work of systematic solutions with experimental studies, and it is still at an immature stage. Challenges may fall on the unobtrusive collection of context and the method of utilising the sensed variables as context to customise the web information to the contextual situation.

(2) Mobile recommendation

The mobile recommendation can be regarded as an essential field of context-aware mobile applications. The application collects the current or history context to predict users’ expectations and provide the users with information or services of interest. Since handheld mobile devices are possessed by almost everyone, it is appropriate to characterise the users and provide computing service of interest to them.

Some mathematical models are proposed and used to obtain appropriate decisions for context-aware recommendation. Yap et al. (2007) proposed a BN-based approach to build a recommendation system that could minimise context acquisition, and case studies on restaurant recommendation and Web page recommendation were conducted. To overcome the missing context input, a two-tiered context model is presented to capture the causal dependencies among context parameters. It is demonstrated that learned BN predicts accurately even when important context inputs are unavailable. Hosseini-Pozveh et al. (2009) proposed a multidimensional approach for context-aware recommendation in mobile commerce. The users, items, the context information and the relationship between them are represented in a multidimensional space. The system determines the usage patterns of each user under different contextual situations and creates a new 2-dimensional recommendation space for final recommendation. Evaluation of this system with restaurant food recommendation considering day, time, weather, and companion as context information shows that the proposed approach increases the recommendation quality. Easylife by Zhuang et al. (2011) analyses users' query history and senses the personalized context to understand users' intent, and then presents it as a ranked list of entities. According to users' implicit intent, the system can recommend the most suitable restaurants in mobile environments.

The mobile recommendation is mainly used to handle information refining, where interesting information is blended with a large amount of irrelevant information. Existing studies focus on the fields of Web information retrieval, music and movie recommendation, restaurant recommendation, and product recommendation, etc. Lee and Park (2007) proposed a mobile Web news recommendation system. The recommended news in this system incorporates news article attributes and user preferences for categories and news articles. User preference is estimated by incorporating news article importance and recency, user preference change, and user preference on news categories and articles. Fong et al. (2011) proposed a Web recommender that models user habits and behaviours by constructing a knowledge base using temporal Web access patterns. The real-life temporal concepts and resources of periodic pattern-based Web access activities are represented and requested with Fuzzy Logic (FL), which is used to provide timely personalised recommendations. Experimental evaluation produces promising results. Yu et al. (2006) introduced a context-aware media recommendation platform to support media recommendation, adaptation, and delivery for smartphone mobile devices. Focusing on the major problem of acquiring and revising user interface for effective recommendation systems, Ricco and Bguyen (2007) designed a product recommendation methodology and implemented in MobuRek, which supported limited asking and answering of questions to help users search travel products based on mobile phones.

It is not difficult to find that performance of context-aware recommendation is mainly determined by two procedures: context acquisition and recommendation decision making. The challenges fall on three aspects: (1) It is difficult to obtain useful context without user interaction; (2) As recommendation systems may incorporate complex mathematical methods, the computation load may be challenging for resource limited devices; (3) The result of context based recommendation is usually closely related to the application domain. It is a critical task to extract the features of information to recommend using the domain knowledge.

(3) Human-mobile interaction

HMI is a newly coined term for human computer interaction of computing systems based on mobile devices. Since integrated, light, and handy mobile devices are inevitably limited in computation power, memory size, communication, and especially user interaction, various built-in or embedded sensing techniques are employed to compensate the limited user interaction of these mobile devices.

Toye et al. (2004) presented an interaction technique allowing users to access mobile service using their camera-phones, public information display, and visual tags. This interaction technique is mainly for the control of site-specific mobile services, and a public information display is used to overcome the limitation of mobile devices' small keypads and screens. The EyePhone system by Miluzzo et al. (2010b) uses the mobile front camera to track the user's eye and map its current position on the display to a function or application. This approach allows the users to launch an application by blinking at an application instead of clicking the mouse. The EyePhone consists of four steps fulfilling a task: eye detection, open eye template, eye tracking, and blink detection. Implementation on a Nokia N710 tablet has been done and results indicate that Eyephone is a promising hand-free manner to drive a mobile application. Lu et al. (2009) proposed a scalable framework named SoundSense for sound events modelling on mobile phones. SoundSense is implemented on the Apple iPhone and is specially designed for resource limited devices. It uses a combination of supervised and unsupervised learning techniques to classify general sound and discover sound events for individual users. Implementation and evaluation of two proof-of-concept applications demonstrate that SoundSense can recognise meaningful sound events of a user's everyday life. Campbell et al. (2012) utilised the neural signals to control mobile phones for hands-free, silent, and effortless human-mobile interaction. The NeuroPhone detects neural signals by mobile phone applications on the iPhone using off-the-shelf wireless electroencephalography (EEG) headsets. A case study on brain-controlled address book dialling up was conducted. The prophase implementation produced promising results for a limited set of cases and many challenges remain unsolved.

From the above investigations, sensing techniques are promising in overcoming the limitation of the mobile devices in user interaction. However, most of the methods are still in lab experiment stage and are not mature for practical use. That is because sensing techniques are susceptible to environmental interference. On the other hand, the sensing techniques may affect usability and efficiency of mobile applications. The interaction methods need to be lightweight enough to guarantee the performance of the applications.

(4) Human behaviour recognition

The behaviour recognition which integrates a user's behaviour and action into the interaction with mobile built-in or wearable sensors is a promising and significant field. It is applicable in many fields such as healthcare, wellbeing, and smart environment user interaction, etc. The mobile devices gather the data and extract the features from the raw data, and then identify the activity with pattern matching or other classification algorithms. Some investigations into human behaviour recognition are introduced as follows.

Due to the convenience of the mobile and wearable sensors, activity recognition has been studied in a variety of application scenarios. A typical case in early stage is single action detection for elderly care, such as falling (Sixsmith and Johnson, 2004; Chen et al., 2005; Hansen et al., 2005) and walking (Pappas et al., 2004; Barralon, 2006) detection. Dai et al. (2010) proposed a platform PerFallD for pervasive fall detection based on mobile phones. Two algorithms are designed for fall detection: one is for acceleration-based detection, and the other is for a magnetic accessory detection based on shape context and Hausdorff distance. Experimental evaluations produce promising results of fall detection and power efficiency.

Huynh (2008) used wearable sensors, mainly accelerometers, to record, model, and recognise human activities focusing on the particular research challenges of the need for less supervision and recognition of high-level activities. This work analyses the features of activity recognition using a set of activities such as walking, standing, sitting, or shopping, and proposes an unsupervised algorithm which can discover structure in unlabelled recording of activities. The weakness of this work lies in the limited use of sensors. So, additional sensors are needed for high-level activity recognition. Choudhury et al. (2008) presented the difficulties in practical applications: the user and environmental conditions, privacy problems, lightweight and unobtrusive sensing technique, trainable machine learning algorithm. An automatic activity recommendation system using on-body sensors is built based on the Mobile Sensing Platform (MSP). Several real world deployments such as UbiFit Garden and recall accuracy were conducted, and the study identified critical capabilities for mobile inference system to support a variety of use scenarios.

Furthermore, there are systems focusing on the recognition of continuous human activity with miniature and low power wearable platforms. For example, Ward et al. (2006) presented a method using a wrist-worn sensor platform with a microphone and accelerometer to detect a set of human activities of a kitchen scenario. In addition to single action detection, the detection moves to multiple actions detection for different purposes. Yang et al. (2009) proposed a distributed recognition framework Distributed Sparsity Classifier for classification of continuous human actions using wearable sensor network. The classification is done in a distributed way on individual sensor nodes and a base station computer. For evaluation, a database called Wearable Action Recognition Database comprising 20 human subjects in 13 action categories was constructed, and results produced promising performance.

From the above investigations, the sensors used for human behaviour recognition are limited to accelerometer, microphone, and magnetic accessory, and most of the systems are based on accelerometer. The mobile device based human behaviour recognition is also in an experimental study stage, which is rarely used in practical applications. As the processing of a continuous signal is executed locally on mobile devices, it consumes a lot of computation power and is challenging for the resource limited devices.

(5) Smart environment

The smart environment which employs the sensing techniques to enhance user interaction with the physical world towards context-aware automation is a promising research area. The applications in this area are designed mainly for home area, meeting rooms, and factories.

The early stage investigations mainly focus on the communication level with the novel wireless modules and network solutions for proof of concept development. Xia et al. (2007) presented an application level design methodology for WSAN in mobile control applications emphasising the packet loss rate with experimental studies. Chung and Oh (2006) presented an indoor environment monitoring solution with well-designed sensor module and RF connection. The indoor vision can be monitored with Web camera, and sensor data is transferred to client PC and PDA for further use. Liang et al. (2008) proposed a wireless smart home sensor network system based on ZigBee and Public Switch Phone Network for indoor network and long-haul network, and presented a discussion of the feasibility for home appliance monitoring and control.

With the development of the relevant techniques, more and more research work has transferred to high-level smart home applications and the models and methods to facilitate occupants' daily life. To deal with the heterogeneity of electronic devices, the DPWS is employed for interoperability using Web service specifications on the resource constrained devices (Chen and Zhi, 2011). Cheng et al. (2012) presented the design and implementation of a service-oriented smart home architecture to integrate popular protocols and coordinate Tmote, Zigbee, and Bluetooth to cover various service-oriented applications in home network and service applications. Bae (2014) presented a method for recognising Activity of Daily Living (ADL) by discovering and monitoring patterns of ADLs in sensor equipped smart homes. It is promising for health monitoring and living assistant applications. Charlon et al. (2013) presented an elderly care system which could identify patients and detect falls with a wearable electronic patch and sensors deployed in different areas of the care unit. A web application is provided for the medical staff to give care, and the results of the system are encouraging. Medjahed et al. (2011) proposed a tele-monitoring system installing sensors at home for elderly people medical care. In integrates elderly psychology and behaviour, environmental conditions, and medical knowledge with a data fusion approach based on FL. In addition, Grønli et al. (2010) presented a smart meeting room through embracing different mobile technologies to allow context-aware abilities in meetings. Korber et al. (2007) presented a design with a highly modular and scalable implementation of WSAN for production automation for factory use.

The recent research work addresses more technical issues around how to implement the techniques to effectively assist people's daily life in smart environments. More and more studies move to dealing with mathematical models for automation decisions, human behaviour modelling and analysis, and high-level knowledge processing, etc.

Although the research and application of context-aware mobile techniques are not limited to the above areas, the applications discussed above reflect the techniques and methods which can be applied in other fields such as mobile navigation, mobile healthcare, mobile learning, and social networks. Therefore, the related work will not be discussed in detail.

From the works reported above, it is found that the context-aware mobile computing has already been a widely concerned research topic. The wide concern is reflected in the following aspects: (1) For many use scenarios, the handheld mobile device with built-in sensors being aware of contextual situation becomes a relatively ideal scheme to enhance the performance of information retrieval and service automation; (2) The applications in the above different areas are progressing in an overwhelming trend; (3) Some PC oriented methods and mathematical models are gradually introduced into the mobile device based context-aware systems. However, the context-aware mobile systems still need time to become mature enough to efficiently handle the context data and produce good performance.

2.4.2 Design Strategy and Architecture Support

As mentioned above, researchers were already aware of the problem of the lack of general framework and considered it as a key problem in context-aware application development (Dey et al., 2001; Chen, 2003; Biegel and Cahill, 2004; Gu et al., 2005). It is evident that proper architectural support is needed for the gathering, modelling, storing, and distributing of context data (Costa, 2003). In order to facilitate mobile-based context-aware application development, a number of design strategies and frameworks are created accordingly focusing on either the frontend context acquisition or the backend data management and service customisation.

The development strategies of context-aware systems are largely rule-based, for which the context-aware information or service is obtained with explicitly defined rules. Daniele (2006) presented a rule-based design approach based on the Event-Control-Action (ECA) pattern, which can manage context information and proactively react to context change. In ECA pattern, the task of gathering and processing context information is the event module, the task of triggering actions in response to context change is the action module, and the application behaviour description controlling the tasks is the control module. The reactive context-aware application behaviours are described using ECA rules with the form *if <condition > then <action>*. ECA is a widely accepted design pattern for context-aware system development. Al-Sammarraie (2011) provided a policy-based approach for context-aware systems, which regarded policy as a major role in constructing the architecture of context-aware systems to control behaviour of the system with well defined rules. From a programming perspective, Zhang et al. (2009) proposed a table-driven programming paradigm which used virtual tables to connect knowledge of both developer and space manager while separating dependency between context and application logic from the base program. This method improves the flexibility and complexity in programming. Rehman et al. (2007) introduced the Model-View-Controller (MVC) for interactive context-aware mobile application development, which aimed to construct a UI layer between users and the backend to communicate the system's inference process. The above strategies for architecture design and application programming can be used to guide context-aware system development for efficiency and simplicity.

In order to provide flexible applications customised to the dynamic contextual situation, infrastructure support is urgently needed. In the early years, such architectures may not be tailored to the special requirements of mobile scenarios regarding the limitations of network connections, power supply, computation power, and memory, etc. With the technical progress of the mobile devices, architectures which provide support for context-aware applications considering the specificity of mobile device and use scenario are proposed.

Early in 2001, a noticeable example framework named Context Toolkit was developed (Dey et al., 2001). The Context Toolkit consists of widgets, aggregators, and interpreters. The widgets gather context information from sensors and provide the interfaces. The aggregators aggregate context information and the interpreters provide the inferred context for applications. In Context Toolkit, the complexity of context retrieval through the use of sensors was hidden by building blocks where all context information was hidden with different components. Context Toolkit is a basic architecture for platform-independent supply of context information. Applications using this framework are fully dependent on widgets and the remote server. The Context Toolkit can facilitate the basic levels of context-aware mobile application development, which is suitable for IF-THEN logic-based computing services. It lacks high-level context handling and further user of context data.

Chen (2003) proposed a Context Broker Architecture to reduce difficulty and cost in building context-aware systems. In this architecture, Web Ontology Language (OWL) is employed for context modelling and privacy policies, Jess is used to build a hybrid reasoning mechanism, and JADE/FIPA are utilised to achieve broker behaviours and agent communications. Chen (2004) presented a context fusion framework named Solar, focusing on fusion of the distributed context data in dynamic computational environment. It is built with a scalable and self-organised service, which allows applications to select distributed data sources and compose them with customised data-fusion operators. This system has taken into account high-level understanding of its execution context, buffer overflow, environment dynamics, and some common failures, and provided corresponding solutions. Gu et al. (2005) proposed a Service Oriented Context-Aware Middleware (SOCAM) architecture for the rapid prototyping of context services. This architecture consists of five independent service components: context provider, context interpreter, context database, context-aware services, and service locating service to efficiently support acquiring, discovering, interpreting, and assessing various contexts to build context-aware systems.

Padovitz et al. (2008) presented a flexible, scalable, and reasoning oriented framework ECORA for context-aware computing based on a functional component library implementing concepts and algorithms of a context space approach. The ECORA is designed focusing on the reasoning of the uncertainty context considering the issues of heterogeneity, scalability, communication, and usability. This framework combines the centralised reasoning service with context-aware and reasoning-capable mobile software agents. Therefore, it provides powerful reasoning capability and provides flexibility and robustness as well. Pung et al. (2009) presented a context-aware middleware infrastructure for pervasive healthcare named Context Aware Middleware for Pervasive Home. This system allows timely and accurate delivery of health or medical information among the patients, doctors, and healthcare workers through widespread deployment of wireless sensors and mobile devices. The key techniques enabling the system are: P2P-based context query processing, context reasoning for activity recognition, and context-aware service management. The main strength of this architecture is its capability to support a context-aware service running over multiple physical spaces.

Shrestha (2010) proposed a robust mobile service oriented architecture framework for building and operating lightweight and flexible context-aware mobile applications. The framework supports different attributes of context information and dynamic integration of context in real time. This work also suggests the problems of this framework: (1) the processing of requests when there is no context database and no metadata in the service registry; (2) implementation details and resources of mobile Web service in the applications; (3) interaction with new mobile devices. Tesoriero et al. (2010) defined a model driven architecture – CAUCE, a model consisting of three layers for context-aware application development. The first layer defines how to build the task, space, and social view of the system. The second layer defines how to build the referential space, the information flow, and the entity context of the system. The third layer defines the deployment environment of the system according to the second layer. In order to clarify the macro-components and their interaction of context distribution, Bellavista et al. (2013) proposed a data-centric architecture model consisting of three principal factors: context data source, context data sink, and context distribution function. The context distribution function contains three main facilities organised in two horizontal layers - context data management and context data delivery, and a cross-layer facility - runtime adaption support. This architecture model is clearly defined, which can be used as a general one to describe context distribution.

From the above discussion, different design strategies are introduced to context-aware mobile system development to handle the events and data or control behaviour of the system with defined policy or rules. On the other hand, it is not difficult to find that dispersion of context source and heterogeneity in techniques are common problems facing architecture design of context-aware applications. Another problem facing context-aware system architecture design is the handling of context data of different semantic levels. In addition, the awareness of quality of the imperfect context data and potential use of history context also needs architecture support for context-aware application development.

2.4.3 Theories and Methods for Computing Service Customisation

To utilise the sensor obtained context data efficiently and provide the appropriate computing service customised to the contextual situation, some mathematical models and theories are required to abstract the sensor data, infer the high-level context, and customise the computing service. In the following, the related mathematical methods for context abstraction and context-aware service customisation decision making are discussed.

2.4.3.1 Context Abstraction Methods

As the raw sensor data may not be used directly by the context-aware application, the context abstraction and normalisation operation is required. To derive the applicable context from the low-level sensor data, several approaches can be employed. The method to use is determined by the attributes of the sensor data and the use case. From the related investigations, the following context abstraction methods are recognised:

(1) Threshold-based method (Heinz et al., 2006)

The threshold-based method is the simplest method for context abstraction, which is appropriate for simple and explicit context data. Provided the variable *x* is the raw sensor data, and *a, b* are the probable value of higher level context, the method can be presented with (2-1).

$$f(x) = \begin{cases} a, & x > Threshold \\ b, & x \le Threshold \end{cases} \tag{2-1}$$

(2) Minimum distance-based method (Könönen and Mäntyjärv, 2008)

Minimum distance-based method is also strong in simplicity and ease of use compared to the complicated mathematical models. It determines the high-level context *S* by calculating its distance with the existing context value C^i, $i=1,2,\ldots,n$ $(n>1)$ denoted by d. The context item with the minimum distance is the particular high-level context. This calculation can be presented with (2-2).

$$d = \min_{i=1}^{n} \left\| S - C^i \right\| \tag{2-2}$$

where $\|.\|$ is a norm such as Euclidean norm and Manhattan norm.

(3) Machine learning method

For some sensors with continuous signal output, the context abstraction may be based on some section of data, such as audio signal for voice recognition. For this situation, many complicated mathematical models are employed for context abstraction, such as BN, FL, Hidden Markov Model (HMM), Decision Tree (DT), k-Nearest Neighbour (kNN), Support Vector Machine (SVM), Artificial Neural Network (ANN), etc. There are also investigations integrating two or more methods to seek high accuracy.

The relevant methods employed in context-aware mobile systems for context abstraction are summarised in Table 2-8. From the table, many methods have been tried for context abstraction in the investigation, and some can produce promising results. Although some of the investigations are done on mobile devices, the computation load of some complicated methods may be a challenge for heterogeneous mobile devices. Research works in this area are not limited to practising the mathematical models in the table. There are also investigations further exploring the feasibility of the complicated mathematical methods, considering their practical application in the mobile computing environment.

Könönen and Mäntyjärv (2008) found that even a simple linear classification algorithm can achieve a reasonably good performance if the features are suitably selected. Yap et al. (2007) found Bayesian Network outperforms J4.8 DT in overcoming both missing and enormous context inputs to generate significantly more accurate predictions for context-aware recommendation. Chernbumroong et al. (2011) employed two classification algorithms DT C4.5 and ANN for human activity recognition using a single wrist-worn accelerometer. Results show that DT C4.5 achieves better performance (maximum 94.13% in four sets of data) than ANN (maximum 90.45% in four sets of data). In addition, Sun et al. (2010) investigated the physical human activity recognition in natural settings where a mobile phone's position and orientation were varying, depending on the position, material, and size of the holding pocket. Promising results were produced with a SVM-based classifier algorithm in this research.

Table 2-8 Context Abstraction Methods

Methods		Investigations	Sensing Techniques	Computation Platforms	Evaluation (Accuracy)
Threshold based method		Motion analysis in martial arts game (Heinz et al., 2006)	Gyroscope and accelerometer	Mobile computer	N/A
		Patient special conditions monitoring (Burchfield and Venkatesan, 2007)	Accelerometer (MMA7260Q)	TI MSP430 Family MCU	N/A
Minimum distance-based method		Human activity recognition (Könönen and Mäntyjärv, 2008)	Acceleration sensor and heart rate signal	Symbian S60	72%-91% Dependent to vote mechanism
Machine learning methods	DT	Activity recognition (Bieber et al., 2011)	Accelerometer and sound sensor	Sony Ericsson Xperia X10 with Android 2.2	Qualitative analysis
		Ambulatory monitoring (Karantonis et al., 2006)	2-axial accelerometer MXR7210GL	Local computer	90.8% (Overall accuracy of 6 subjects)
		Physical activity detection (Bao and Intille, 2004)	2-axial accelerometer ADXL210E	Mobile phone	84%
	kNN	Real-time monitoring of human activity (Brezmes et al., 2009)	Accelerometer	Nokia N95	70%
		Activity recognition (Ravi et al., 2005)	3-axial accelerometer CDXL04M3	HP iPAQ	80%
		Human activity recognition (Lombriser et al., 2007)	SensorButton - 3-axis accelerometer, MEMS microphone, light sensor	SensorButton with MSP430F1611MCU	91%
	SVM	Activity recognition (Sun et al., 2010)	Accelerometer	Mobile phone	93.1% for varying position 94.8% for fixed position
		Human activity recognition (He et al., 2008)	3-axis accelerometer	Mobile phone	92.25%
	BN	Activity classification (Saponas et al., 2008)	3-axis accelerometer	Apple iPhone	97%
		Context aware recommender (Yap et al., 2007)	Context inputs	Service end	Outperforms J4.8 DT
	HMM	Biometric gait recognition (Nickel et al., 2011)	Accelerometer	G1 mobile phone with Android	FNMR: 10.42% FMR: 10.29%
		Activity recognition (Lee and Cho, 2011)	Accelerometer	HTC Desire with Android	Compared HHMM, HMM, and ANN
		Activity recognition (Martin, 2011)	Wireless accelerometer	Motorola Droid	90% (Average)
	GMM	Posture and movement classification (Allen et al., 2006)	2-axial accelerometer	Backend	91.3% (Average)
		Detecting phone context (Miluzzo et al., 2010c)	A set of sensors (light sensor and accelerometer)	Nokia N95 & Apple iPhone	80%
	FL	Real-time fall detection (Dinh and Struck, 2009)	Single accelerometer	PC	94%
		Activity recognition (Yang et al., 2007)	3-axial accelerometer MMA7260Q	N/A	83.41±5.93% (FSS method) 92.86±5.91% (LDA method)
	ANN	Human activity recognition (Khan et al., 2010)	3-axial accelerometer	Smartphone	96% (Average)
		Motional activity recognition (Györbíró, 2009)	Motion Band sensors	Desktop workstation & Nokia 6630	79.76% (Average)
		Physical activity monitoring (Jafari et al., 2007)	Sensor Mode	Laptop PC & Nokia 6680	84% (Average)

For evaluation, some investigations use mean value, standard deviation, Root Mean Square (RMS), etc. Most of the works use accuracy rate or error rate to assess the performance of the selected algorithms. Nickel et al. (2011) used the False Non-match Rate (FNMR) and False Match Rate (FMR) for evaluation which could better evaluate the selected methods.

From the above investigations, it is not difficult to find that context abstraction for mobile device based systems has its specificities compared to the method for traditional platform based systems:

- Computation power of mobile devices is limited and it varies between devices, so resource consumption of algorithms should be taken into account.
- In the studies, the abstraction methods are specialised for particular applications in offline mode; to consistently achieve the methods online for practical use is challenging.
- The performance of context abstraction is multi-faceted, which is easily affected by ambient environment, people involved, devices employed, length of sample data, etc.

The above issues should be considered for context abstraction with regard to the mobile and pervasive computing environment. Because of the heterogeneity and resource limitation in devices, lightweight and efficient algorithms are expected for practical mobile based context-aware systems.

2.4.3.2 Decision Making for Context-aware Service Customisation

With the applicable context data, the system can then provide the user with the computing service that is customised to the contextual situation. As the ultimate goal of the context-aware system is to promote the efficiency and usability of the applications and provide the user with expected computing service, the method utilising the context data to create service customised to the contextual situation is important to the system.

The traditional approach to deal with the application adaptation is based on IF-THEN rules (Rarău et al., 2005). As the IF-THEN rules are independent of each other, the context change must be specified for each action for context-aware application. With the emergence of rich sensing devices, the increase in quantity of context and the complexity of its relationship requires more powerful approaches dealing with the new situations. For the new context-aware computing systems, some theories are introduced in this area for context-aware service decision making, such as Fuzzy Multi-Attribute Decision Making (FMADM), Multi-facet Item based method, Multi-Attribute Utility Theory (MAUT), Adaptive Neuro-Fuzzy

Inference System (ANFIS), Rough-Fuzzy method, and probability-based model. This section will provide an in-depth discussion of these theories and methods in dealing with context-aware services and applications.

(1) Fuzzy Multi-Attribute Decision Making

In order to allow the computing system to make appropriate decisions on behalf of users according to dynamic user context, TalebiFard and Leung (2011) introduced a FMADM method and a context similarity measurement method, which is explained as follows.

Let $\tilde{a}_i$ be fuzzy triangular numbers defined as: $\tilde{a}_i = (a_i^l, a_i^m, a_i^u)$, where *l*≤*m*≤*u* and *l* and *u* are the lower and upper values.

Weight normalised decision matrix is calculated as:

$$\tilde{v}_{ij} = \tilde{w}_j . \tilde{a}_{ij} / \sqrt{\sum \tilde{a}_{ij}^2} \tag{2-3}$$

where i=1,2,…,m, j=1,2,…,n, and $\tilde{v}_{ij} = (v_{ij}^l, v_{ij}^m, v_{ij}^u)$.

Find the best and worst solution according to benefit and cost criteria with:

$$A^* = (\max_i \tilde{v}_{ij} \mid j \in J) \wedge (\min_i \mid j \in J^{'}), i = 1, 2, ..., m \tag{2-4}$$

$$A^- = (\min_i \tilde{v}_{ij} \mid j \in J) \wedge (\max_i \mid j \in J^{'}), i = 1, 2, ..., m \tag{2-5}$$

Then, find the Euclidean separation of each alternative from the best and worst solution:

$$s_i^* = \sqrt{\sum_{j=1}^n (\tilde{v}_{ij} - \tilde{v}_j^*)^2}\text{, i=1,2,…,m.} \tag{2-6}$$

$$s_i^- = \sqrt{\sum_{j=1}^n (\tilde{v}_{ij} - \tilde{v}_j^-)^2}\text{, i=1,2,…,m.} \tag{2-7}$$

With s_i^* and s_i^-, calculate the relative closeness to the best solution, and rank according to the preference order.

$$c_i^* = s_i^- / (s_i^* + s_i^-) \tag{2-8}$$

where $0 < c_i^* < 1$, i=1,2,…,m.

Then, a α – level set of fuzzy set M is defined for the ranking of fuzzy preference:

$$M_\alpha = \{x \in U \mid \mu_M(x) \geq \alpha\} \tag{2-9}$$

For the decision making, the measurement of context similarity between user and service is:

$$Sim(c_i, c_j) = \eta[\sum_{k=1}^N w_k (a_i - a_j)^2]^{1/2} \tag{2-10}$$

where $w_k \in (0,1)$ is the weight for each attribute value, $\sum_{a_i \in C} w_{a_i} = 1$, and η is the feature similarity which is defined as:

$$\eta = |F(c_i) \cap F(c_j)| - \alpha |F(c_i) \Delta F(c_j)| \qquad (2\text{-}11)$$

where α is a constant indicator of penalising alternatives with more distinct features.
This method can be used as a generalised approach to collect the context of mobile devices, and compare the context with the advertised service based on the feature similarity. The fuzzy set is appropriate for presenting the uncertainty of context data, and the similarity measurement provides a way to evaluate the context and the service conditions, while the capability handling context data with complicated structures and relationships needs further verification.

(2) Multi-Facet Item based method

Rarău (2006) proposed a multi-facet item abstraction method as an adaptation mechanism in his PhD study, which uses the facet to contain all the elements as context that can be accessed by applications or users. The Multi-facet item adaptation model is as shown in Figure 2-2.

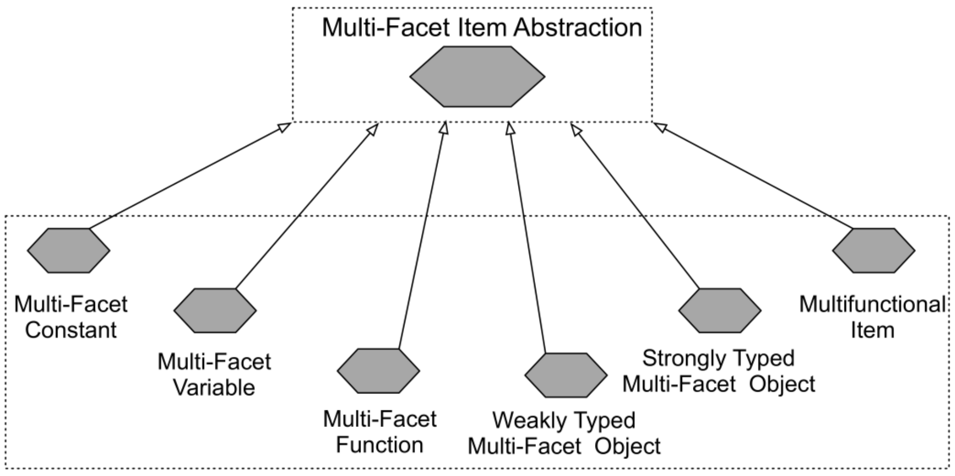

Multi-Facet Item Abstraction
Multi-Facet Constant
Multi-Facet Variable
Multi-Facet Function
Weakly Typed Multi-Facet Object
Strongly Typed Multi-Facet Object
Multifunctional Item

Figure 2-2 Multi-Facet Item Adaptation Model (Rarău et al., 2006)

Let $S = \{S_1, S_2, ..., S_n\}$ be the set of services provided by the multifunctional item and let $F = \{F_1, F_2, ..., F_m\}$ be the set of facets. The facet is represented as an n-tuple:

F=(context, {e|e∈C},state,t,accessibility,show, hide, priority)

where *state* can be *active* and *inactive,* and *accessibility* can be values *exposed* and *hidden*.

The modification of the facet state is described with the updating function:

$$\alpha : \mathrm{F}_C \times T^{ev} \rightarrow \mathrm{F}_C \qquad (2\text{-}12)$$

An exposing strategy is made by the mechanism managing the accessibility of application or users. A multifacet item consists of a multifact collection and an exposing strategy can be presented by:

$$multifacetItem=(C,ExposingStrategy)$$

The item behaviour *S* is defined by the algorithm for reaction on context changes and the algorithm for adjusting the context-awareness.

This work focuses on the modelling and usage of context as a context-aware computing service strategy, with multi-facet item based framework and toolkit built with the strategy. The context-aware service adaptation may be implemented in some functional modules of particular applications which fall outside the focus of this work.

(3) Multi-Attribute Utility Theory

The MAUT is widely used for product evaluation in consumer organisations. It is also an applicable method for context-aware service customisation. Schäfer (2001) employed the MAUT to estimate the user’s interests for product recommendations. The evaluation *v(x)* of an object *x* is defined as a weighted addition with respect to its relevant value dimension by function:

$$v(x)=\sum_{i=1}^{n} w_i v_i(x) \tag{2-13}$$

where $v_i(x)$ is the evaluation of the object on the i^{th} value dimension d_i, w_i is the weight of the value dimension, *n* is the number of different value dimensions, and $\sum_{i=1}^{n} w_i = 1$. For each value dimension d_i, the evaluation $v_i(x)$ is defined as the evaluation of relevant attributes by:

$$v_i(x)=\sum_{a\in A_i} w_{ai} v_{ai}(l(a)) \tag{2-14}$$

where A_i is attributes set of d_i, $v_{ai}(l(a))$ is the evaluation of the actual level *l(a)* of attribute *a* on d_i, w_{ai} is weight indicating the impact attribute *a* on value dimension d_i in the evaluation, and $\sum_{a\in A_i} w_{ai} = 1$.

For context-aware systems, the computing service can be regarded as the evaluation *v(x)*, the relevant context information can be considered to be the attribute a_i, and weight w_{ai} is the impact of context to the decision for the computing service. This method is easy to use and the logic in computation is explicit. However, it is difficult to determine the weight of the attributes of the context items in practical use. Moreover, the method is not strong enough to present the complicated relationships of the service and context items for some use scenarios.

(4) Adaptive Neuro-Fuzzy Inference System

The ANFIS was introduced to a mobile-learning system by Al-Hmouz et al. (2012) for adaptive learning content distribution, which aimed to adapt the learning content to learners' need for different learning context scenarios. ANFIS uses FL to transform given inputs into a desired output through highly interconnected Neural Network processing elements and weighted information connections. The reasoning model of ANFIS is as shown in Figure 2-3.

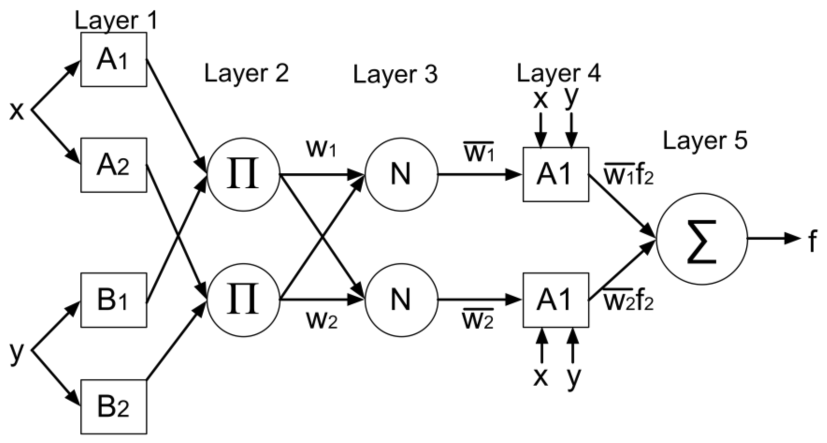


Figure 2-3 ANFIS Model (Al-Hmouz et al., 2012)

Two fuzzy IF-THEN rules are used to present the ANFIS model:

Rule (1): IF x is A$_1$ AND y is B$_1$, THEN $f_1 = p_1 x + q_1 y + r_1$

Rule (2): IF x is A$_2$ AND y is B$_2$, THEN $f_2 = p_2 x + q_2 y + r_2$

In the rules, *x* and *y* are the inputs, A_i and B_i are the fuzzy sets, f_i are the outputs within the fuzzy region specified by the fuzzy rules, and p_i, q_i, and r_i are the design parameters that are determined during the training process.

In Figure 2-3, a circle denotes a fixed node and a square denotes an adaptive node. The ANFIS is a five-layer architecture, which is explained as follows.

The outputs of Layer 1 are the fuzzy membership grade of the inputs calculated by equations:

$$O_{1,i} = \mu_{Ai}(x),\ i=1,2. \tag{2-15}$$

$$O_{1,i} = \mu_{Bi-2}(y),\ i=3,4. \tag{2-16}$$

where *x* and *y* are inputs to the node *i*, and *Ai* and *Bi* are the linguistic labels associated with the node function. $\mu_{Ai}(x)$ and $\mu_{Bi-2}(y)$ can adopt any fuzzy membership functions.

Layer 2 involves fuzzy operations, which uses the *AND* operator to fuzzify the inputs. It behaves as a simple multiplier and is labelled with *Π*. Layer 3 plays a normalisation role to the firing strengths from the previous layer, which is labelled with *N*. The output of nodes in Layer 4 is simply a product of the normalised firing strength and a first order polynomial. Layer 5 performs the summation of all incoming signals labelled with *Σ*, and the overall output of the model is obtained. Outputs of layer 2 to layer 5 are given by equations as below:

$$O_{2,i} = w_i = \mu_{Ai}(x) * \mu_{Bi}(y)\text{, i=1,2.} \tag{2-17}$$

$$O_{3,i} = \tilde{w}_i = w_i / (w_1 + w_2)\text{, i=1,2.} \tag{2-18}$$

$$O_{4,i} = \tilde{w}_i f_i = \tilde{w}_i / (p_i x + q_i y + r_i)\text{, i=1,2.} \tag{2-19}$$

$$O_{5,i} = \sum_i \bar{w} f_i = (\sum_i w_i f_i) / \sum_i w_i\text{, i=1,2.} \tag{2-20}$$

ANFIS integrates the FL and Neural Network, which can tune the parameter of Fuzzy Inference System (FIS) with Neural Network learning. It refines the fuzzy IF-THEN rule for complex context-aware applications. This method is strong in the capability of dealing with the incompleteness of rules made by human expert with fuzzy rule training. It also eases the implementation of the system and enables fast and accurate learning.

(5) Rough-Fuzzy method

Duan et al. (2007) proposed a rough-fuzzy hybridisation for preference-based Web information retrieval. In this rough-fuzzy method, fuzzy sets are used to handle real-valued weight in the document, and the rough-fuzzy method named Variable Precision Rough Set Model (VPRSM) is used to discover user preference.

The fundamental notation of the VPRSM is the generalisation of the standard inclusion relation called major inclusion relation, which is defined as:

$$c(X,Y) = \begin{cases} 1 - card(X \mid Y) / card(X), & if\ card(X) > 0 \\ 0, & if\ card(X) = 0 \end{cases} \tag{2-21}$$

where *X* and *Y* are subsets of the universe *U*. The majority inclusion denotes the relative degree of misclassification of set *X* with regard to set *Y*.

Let $X \subseteq U$, *R* be an equivalence relation to *U*, the $\beta - lower$ and $\beta - upper$ approximation of *X* can be defined as:

$$\underline{R}_{\beta}X = \{x \in U : c([x]_R, X) \leq \beta\} \tag{2-22}$$

$$\overline{R}_{\beta}X = \{x \in U : c([x]_R, X) < 1-\beta\} \tag{2-23}$$

Then, the definition of $\beta - positive\ region$, $\beta - negative\ region$, and $\beta - boundary\ region$ based on VPRSM are defined as:

$$POS_{R,\beta} = \underline{R}_{\beta}X \tag{2-24}$$

$$NEG_{R,\beta} = U - \overline{R}_{\beta}X \tag{2-25}$$

$$BND_{R,\beta} = \overline{R}_{\beta}X - \underline{R}_{\beta}X \tag{2-26}$$

Further, let *P*, *Q* be equivalence relation on *U*, *P-positive region* of *Q* and dependency function are defined as:

$$POS_{P,\beta}(Q) = \underset{X \in U/Q}{U} \underline{P}_{\beta}X \tag{2-27}$$

$$r_{P,\beta}(Q) = | POS_{P,\beta}(Q) | / | U | \tag{2-28}$$

Based on the VPRSM, a quick reduct algorithm is used to generate the reduct of keywords, and a discerning keyword extraction algorithm is used to extract a minimal set of maximally discerning keywords from keywords set according to users' rating. The user's preferences and discerning keywords are analysed to refine a query to better represent the user's interests, and retrieved results are re-ranked according to rough similarity measures between the refined query and documents.

The fuzzy sets are used to refine the representation of term vectors through discretising the weights for each term to five fuzzy levels "*Very Low*", "*Low*", "*Moderate*","*High*","*Very High*". The relevance degree is defined as:

$$relevance\_\deg = \sum_{i=1}^{n} w_i \times (n-i+1)/n \tag{2-29}$$

where *n* denotes the top *n* pages from the URL list, *i* denotes the i^{th} page in the results, and w_i represents the weight of page *i* manually chosen from 1 and 0 based on the relevance.

This rough-fuzzy method integrates rough set with fuzzy set, which are appropriate to deal with important tasks of personalised Web information retrieval: weight evaluation, ambiguity of language handling, and preference discovery. However, a particular training process for each user seems indispensable, where automatic methods may be expected in practical use.

(6) Probability-based model

Wang et al. (2012) presented a probability-based model for music recommendation based on context information and music content analysis, exploiting the rich sensing capability of mobile devices. The probability-based model is explained as follows.

Let *S* be a set of music recommendation services and *C* be a set of context categories, vector *f* is the extracted feature of sensed data. By Bayes' rule, probability distribution $p(c \mid f)$ is

$$p(c \mid f) = \frac{p(f \mid c) p(c)}{p(f)} \propto p(f \mid c) p(c) \tag{2-30}$$

Then, a random variable $R \in \{0,1\}$ means the suitability of service *s* for context *c*, and the probability $p(R=1 \mid c,s)$ indicates the user satisfaction degree of *s* in context *c*. The joint probability is:

$$p(c, f, R, s) \propto p(f \mid c) p(R \mid c,s) p(c) \tag{2-31}$$

Given the feature vector *f* abstracted from sensor data, the determination of service *s* that maximises the user satisfaction $p(R=1 \mid s, f)$ can be calculated by accumulating the jointly probability of all possible context categories:

$$p(R=1 \mid s, f) \propto p(R=1, s, f) = \sum_{i=1}^{|C|} p(R=1, s, f, c_i) \tag{2-32}$$

To model the implicit feedback from users, a probability prior $beta(\theta, a, b)$ is assigned to $p(R \mid c,s)$ for every (c,s) pair, where *a* and *b* are the quantity of negative and positive feedbacks. The probability $p(R=1 \mid c,s)$ can be expressed as equation:

$$p(R=1 \mid c,s) = b / (a+b) \tag{2-33}$$

User feedback can be defined with a triple $x = (f, s, r)$, where *r* is the observation of the user feedback *R* (0 for negative feedback and 1 for positive feedback). The estimation of *c* denoted by $\hat{c}$ is:

$$\hat{c} = \arg\max_{C} p(c \mid f) \tag{2-34}$$

The corresponding probability prior for pair $(\hat{c}, s)$ is updated to:

$$p(\hat{\theta} \mid x) = \begin{cases} beta(\hat{\theta}, a+1, b), r=0 \\ beta(\hat{\theta}, a, b+1), r=1 \end{cases} \tag{2-35}$$

Then, the corresponding $p(R=1 \mid \hat{c}, s, x)$ representing the user's preference is updated to:

$$p(R=1 \mid \hat{c}, s, x) = \begin{cases} b/(a+b+1), & if\ r=0 \\ (b+1)/(a+b+1), & if\ r=1 \end{cases} \tag{2-36}$$

Finally, the estimation $\hat{c}$ is obtained, and then counters *a*, *b*, and $p(R=1|\hat{c},s)$ are updated.

By monitoring the user's operations, user preference is updated in real-time to adapt to particular users. This probability-based method is lightweight in computation and is easy to implement. It can adapt to new users immediately without a complicated training process. It is appropriate for the context-aware systems without too many context items taking effect.

The major concerns of these mathematical models are summarised in Table 2-9 as below.

Table 2-9 Mathematical Models for Context-aware Service Customisation

Mathematical model	Purpose or use case	Context used	Evaluation method	Strength and weakness
FMADM (TalebiFard and Leung, 2011)	General methodology for context-aware service delivery	Network attributes: bandwidth, bandwidth variation, availability, stability, error rate	Through case study on improving QoE for DSL services, such as VOIP and IPTV	The method is general for context-aware systems QoS is taken into account Not strong in dealing with context with complicated relationship
Multi-Facet Item based method (Rarău, et al.,2005;Rarău ,2006)	General method for context-awareness	Not specified	The overhead of updating the current facet items is measured	The method is general for context-aware systems The adaptation mechanism coordinates the triggering of actions Not strong in dealing with context with complicated relationship
MAUT (Schäfer, 2001)	Estimation of user's interest for product recommend ation	Attributes of the product	Brief case study without evaluation	Simple and easy to implement Difficult to determine the weight of attributes Not strong in dealing with context with complicated relationship
ANFIS (Al-Hmouz et al., 2012)	Adaptive learning content distribution for mobile learning	Learning materials, personal information, location, time, mobile device, environment, network, bandwidth	The performance of ANFIS is evaluated using standard error measurement, which reveals the optimal setting necessary for better predictability.	It integrates fuzzy inference with neural network Strong in the capability of dealing with the incompleteness of rules with fuzzy rule training. The structure is complex
Rough-Fuzzy method (Duan et al., 2007)	Personalised Web information retrieval	User preference and keywords of documents	The method is evaluated by comparing the results of proposed approach with results of Google search	Rough sets and fuzzy sets are integrated for the adaptation. The fuzzified weight and rough similarity are appropriate for context based decision making A training process is required.
Probability based model (Wang et al., 2012)	Mobile phone music recommend ation for daily activities	User activity and music content	Model accuracy and system usability are evaluated by experiments on Android devices	Successfully solved the cold-start problem by combining music content analysis and activity inference The adaptation can be done directly on mobile phone without use of backend server

In addition to the above methods, there are investigations striving to optimise the performance of context-aware mobile systems. For instance, Yap et al. (2007) justified the necessity of minimising the context acquisition due to the uncertainty in mobile and pervasive environments, which involve missing and erroneous context inputs. Then, a Bayesian Network based approach is proposed to obtain the minimal context items that are of interest to the users.

From the above discussion, the high-level supervision for context-aware service customisation with the context data is an effective way to pursue user satisfaction with the computing service. Traditional context-aware mobile systems are based largely on IF-THEN logic, and their capability in dealing with complex contextual situations is limited. Thereby, some mathematical models are presented accordingly. These new mathematical models are more powerful in handling the diversity of context data in complex use scenarios. Although a lot of progress has been made, there are remaining problems worth further investigation. Firstly, it is still a challenge to manage the sophisticated relationship between the entities in some complex context-aware systems. Secondly, the existing methods mainly focus on service adaptation with contexts of interest, the following aspects are seldom discussed: (1) how to determine the context items related to some context-aware service; (2) how the weight of the context items are obtained; and (3) whether the methods employed are lightweight enough for mobile platforms limited in computation and communication power.

2.5 Problems Identified

From the above discussion, context-aware mobile systems have already attracted much research interest and relevant investigations have already spread into many areas. However, it is still at an immature stage and is constrained by some technical challenges. The main problems identified that face the mobile devices based context-aware systems are:

(1) Heterogeneity of techniques and lack of generality in architecture support

The heterogeneity in sensing techniques, mobile devices, and wireless infrastructure raises challenges for context acquisition, data representation, and distribution of context between different devices or function modules in context-aware systems. Therefore, general and interoperable context-aware system architecture is expected to deal with these problems. However, it lacks general architecture that can deal with the heterogeneity problem and provides high-level supervision for context-aware computing systems.

(2) Unobtrusive context acquisition and abstraction with resource limited mobile devices

Admittedly, some context data is not easy to obtain for the context-aware systems. It is also a challenging task to guarantee the quality and stability of context with resource limited devices in a wireless environment. The context information is used to minimise human efforts in the use of context-aware applications. However, the acquisition of context may complicate the operations and context abstraction may overload resource limited mobile devices. Therefore, the unobtrusive, lightweight, and efficient acquisition and abstraction of context is challenging but significant for context-aware systems.

(3) The gap between raw sensor data and applicable context values

Due to the diversity, uncertainty, and incomplete nature of sensor data, the QoC may deeply affect the performance of context-aware applications. Therefore, it is a critical task to evaluate the QoC and effectively use the QoC parameters to promote performance of the applications. On the other hand, the context gathered with sensors is usually low-level context. But the context usable by the applications may be high-level context that can describe the contextual situation comprehensively and accurately.

(4) High-level supervision for computing service customisation

For most context-aware systems, computing service decision making is based largely on the simple rule-based logic which may require much pre-definition and user attention. The methods are weak in dealing with the complex relationship of context to provide the appropriate computing service. The lightweight requirement may be another challenge for resource limited devices. On the other hand, the system lacks the capability of taking advantage of the history context to generate new rules to supervise the computing service customisation.

This investigation will focus on the above research questions to build a system architecture first, and then seek the appropriate approaches to unobtrusively gather the context data, efficiently abstract the useful features, and interoperably distribute the context data in the system. In addition, context-aware service customisation strategy and rule generation mechanisms are investigated to minimise human supervision and promote the usability and efficiency of context-aware mobile applications.

2.6 Summary

This chapter gives an overview of the related research areas and provides some insight into context-aware mobile computing systems. It starts with a discussion of the definition of context and context-awareness. The relevant applications, context-aware framework, and mathematical methods are then discussed. It is found that the heterogeneity of techniques and lack of architecture support, unobtrusive context acquisition, gap between sensor data and applicable context, and high-level supervision of computing service customisation are the critical challenges facing mobile device based context-aware systems, which fall in the focal research area of this investigation.

Chapter 3 Aim, Objectives, and Methods

With an overview of the enabling techniques, related works, and methods for service customisation through the literature review in Chapter 2, the challenges facing context-aware mobile computing are then identified. This chapter puts forward the aim and objectives of this investigation, and the research method employed to achieve the aim and objectives is presented with explicit steps that are mapped to the specific work in the following chapters.

3.1 Aim and Objectives

Progress of the enabling techniques and maturity of theoretical methods in the relevant fields are promising to support practical application of context-aware systems in reducing the human efforts for some use cases. It is significant to identify the relevant challenges and make genuine progress to fill the gap between promising techniques and the practical lagging application of sensor-based context-aware mobile systems.

This investigation aims to chart the technical solutions and theoretical models to establish a general context-aware system architecture which integrates the context information into computation to provide a proactive computing service customised to the dynamic contextual situation. The target context-aware system can then reduce human efforts in the operation of applications by the supervision of context-based service provisioning and its learning mechanism. Practically, the widely used smartphone devices and commercially available low cost sensors are promising media that can be used to characterise the context of the computing tasks. The essential points to achieve the aim may fall on two critical aspects: a general and interoperable system architecture to efficiently handle the context data, such as acquisition, updating, querying, storage, and distribution; and a context-aware service mechanism that can create appropriate computing service according to the context with the self-learning ability to adjust the methods creating the computing service.

This work is different from most of the peer studies that focus on the new techniques or particular applications only, in that it pays more attention to general architecture support, the models and methods creating the context-aware service, and the way to facilitate a service better customised or adapted to dynamic and complicated contextual environments. To achieve the aim of this investigation, the objectives of this research are listed as follows:

(1) To model the context-aware computing system and establish a general, flexible, and reusable context-aware system architecture for context-aware system development.

(2) To examine available context in the real world enabled by mobile built-in and embedded sensors and provide methods for gathering, abstracting, and representing context data.

(3) To provide appropriate context modelling and representing approaches in the interest of interoperability and efficiency in context distribution and context reasoning.

(4) To design a context-aware service provisioning strategy to deal with the diverse context and customise the computing service to the contextual situation.

(5) To take advantage of the history context to discover potential rules to supervise the service provisioning for better customisation of the computing service.

The research work is carried out towards a target system that reaches the objectives addressed above which have covered the essential points to achieve the purpose of this investigation. The emerging techniques, theoretical models, and mathematical algorithms are critical in handling the context data and improving the efficiency of context-aware service provisioning. From a practical perspective, a prototype system is built and case studies are implemented as proof of concept demonstration. Lessons are learned in the system architecture design, context acquisition, context modelling, context-aware service provisioning, context-based rule generation, and these works outline the roadmap of this research.

3.2 Research Methods

This research work involves several research fields and a variety of techniques, such as embedded technology, wireless sensor networks, wireless communication, mobile computing, computing service, and knowledge discovery. When it comes to the context-aware system in practical use, the heterogeneity in techniques, complexity of context environment, uncertainty of context data, and human involvement make the investigation very complicated. Therefore, the methodology of the research determines how reasonably the research can be carried out and whether the findings can be taken seriously.

Generally, a scientific research can be considered as an activity contributing to the domain knowledge within a given field, and the process of research often includes a systematic investigation into a problem or a domain to acquire new knowledge, establish facts, prove ideas or propose new theories (Grønli, 2012). The experimental data obtained is fundamental for result evaluation and new knowledge discovery. The process of this investigation follows the same basic structure of the scientific research: problem definition, literature review, aim and objectives justification, design and evaluation, discussion and conclusion, though slight differences may exist.

In scientific and social research, positivist and interpretive paradigms are commonly used research perspectives. A positivist research focuses on science as a product and results are gathered through means of measurement, quantification, hypothesis testing, and inference from a sample to a stated population (Orlikowski and Baroudi, 1991). This investigation strives to provide a proactive and automated service with a context-aware service architecture to customise the computing service with the ambient context data. The essence of the work is to reduce or eliminate user efforts in the computing service to assist people’s daily life with the novel sensing techniques. Quantitative methods including surveys, experiments, and numerical methods such as mathematical modelling will be employed. Qualitative analysis is also used in the comparison and discussion of the techniques and methods employed in the design and implementation. The objective of these methods is to obtain objective, independent, and generalised contributions in the particular research area. In what follows, the research methodology applied to achieve the research objectives is detailed and justified.

3.2.1 Design Science Research (DSR) Methodology and this Investigation

The DSR is a methodology defined as learning through building, which has already had extensive use in the fields of computer science and engineering. In the information system field, the DSR has become a field rapidly evolving in recent years. Researchers argue that DSR is a well-grounded and well-founded process to use as the research methodology (Hevner et al., 2004). Vaishnavi and Kuechler (2004) suggested a framework for the DSR methodology, as shown in Figure 3-1.

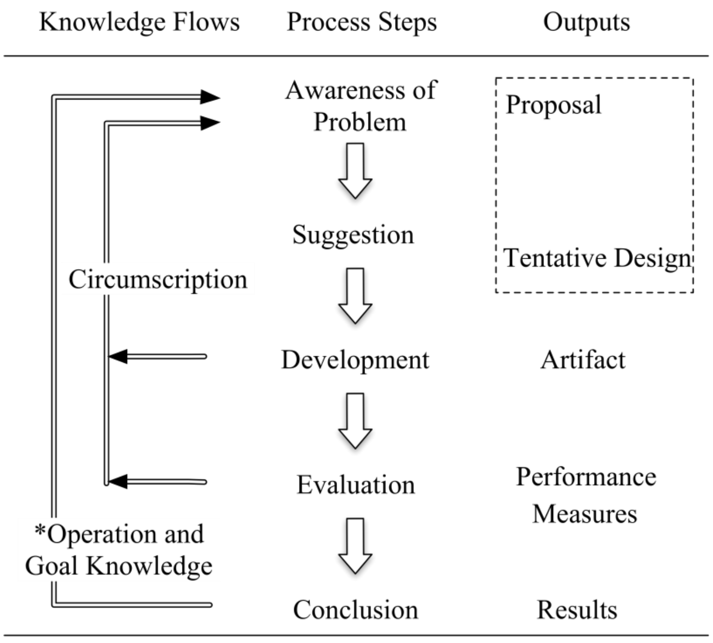


Figure 3-1 General Framework of DSR Methodology

This research is based on the positivist paradigm, and is concerned with problem solving tasks within the fields of electronics engineering, information and communication technology. As design science research is fundamentally a problem-solving methodology with research based on construction and evolution of artefacts (Hevner et al., 2004), this approach is well-suited to meet the aim and objectives of this thesis. The work presented in this thesis is carried out following the DSR methodology, and the content of this thesis is mapped to the flow of DSR methodology as shown in Table 3-1.

Table 3-1 Design Science Research Process and this Study

DSR Steps	Steps in the DSR Process	Chapters in this Thesis
Awareness of problem	Identify the problems in the research domain, and define the challenges in fixing the problem	Chapter 2: Literature review
Suggestion	Research hypothesis is proposed, the aim and objectives of the investigation are raised	Chapter 3: Aim, objectives, and methods
Development	Design framework, choose appropriate techniques to implement the proposed system/application to verify the hypothesis of research. Apply case studies, experiments, tests, and other appropriate activities to demonstrate how the problems identified are fixed.	Chapter 4: Context-aware Computing architecture Chapter 5:Context modelling and context acquisition Chapter 6:Context computation and service provisioning Chapter 8:Case studies design and implementation
Evaluation	Measurement and evaluation using mathematical methods and relevant parameters. Both quantitative evaluation and qualitative comparison and discussion are carried out.	Chapter 7: Context-aware mobile system evaluation framework Chapter 8:Case studies and evaluation
Conclusion	The problem definition, suggestion, development, and evaluation are integrated to justify the product and contribution of the research.	Chapter 9: Thesis conclusion

Following the DSR work flow, in the first place is the awareness of the research question and an in-depth discussion in the literature review. According to the problem definition, the aim and objectives are raised. Then, design and implementation is conducted with regard to the technical challenges to reach the purpose of the research. With the system designed, the evaluation is carried out with quantitative and qualitative approaches. Finally, products and conclusions are justified based on the design, implementation, and evaluation.

3.2.2 Devices, Technologies, Platforms

In this section, the devices, technologies, and platforms employed to build a context-aware system are introduced within the scope of described research methodology. The devices used may include sensors, embedded controllers, wireless transceivers, and the mobile devices with different mobile operating systems to build the pervasive computing environment. In addition, the wireless infrastructure for sensor data acquisition and user interaction, and the service platform providing the context-aware service are briefly introduced.

(1) Devices employed

- Sensors and actuators

  The sensors are the electronic devices that are used to gather the context data. Some of the context from the physical environment such as the temperature, light brightness, humidity, smoke, barometer, and noise level are collected with physical sensors. Of course, the mobile devices also contain some built-in sensors such as camera, microphone, accelerometer, ambient light sensor, etc.

  Actuators are the devices that can be used to control a facility with the context-aware service as the command. Relay and motor are commonly used actuators.
- Embedded controller

  The embedded controller such as MCU, Digital Signal Processor (DSP), and Advanced RISC Machine (ARM) collects the sensor data and transmits the sensor data into the wireless network through wireless transceivers. It can also do some pre-processing on the raw sensor data to improve the quality of the obtained context data.
- Wireless transceiver

  The wireless transceiver converts the signal from one form to another which enables the integration of different communication interfaces into the context-aware systems. For instance the WiFly RN-370M converts the signal from RS-232 serial signal to WiFi signal. Thus, the low-layer electronics can then be connected to the WiFi network (Roving Networks, 2011a). Similarly, RN-274M converts the RS-232 serial data into the Bluetooth network (Roving Network, 2011b).
- Mobile devices

  The mobile devices are very important as they can be used to characterise users' individual preferences and personal context. They also behave as a user interface for the interaction between the users and the computing system. The devices used in this investigation are the latest smartphone devices with cameras, full-touch screens, and internet connections. Most of the smartphone mobile devices ship the mainstream mobile operating systems, such as Apple iOS, Google Android, Microsoft Windows Phone 7, Nokia Symbian, and RIM BlackBerry.

(2) Wireless network infrastructure

In order to facilitate the users with the context-aware system, the devices are embedded into people's lives in an unobtrusive way. Thus, the context data acquisition and service provisioning are performed in a wireless environment. In order to reduce the cost and development cycle, the publicly available wireless infrastructures are promising candidates. There are two promising candidate public wireless infrastructures: WiFi and 3G.

- WiFi – a public wireless infrastructure which is available in buildings and is appropriate for home and office use.
- 3G – a public wireless network that is not limited to in-building areas and is appropriate for outdoor use.

In the wireless network, there are TCP/UDP socket connections and HTTP connections between the devices. These kinds of wireless connections are used in the smart home case study, which employs diverse context to automatically control the home facilities.

(3) Context-aware service platform

In the proposed context-aware mobile computing framework, the context-aware service is provided with context-aware service end. The context information is collected and stored in the database, and service is created with the machine learning technique based on the current context, history context, and user interactions automatically or with human supervision.

- Service application - Service architecture is built to provide context-aware computing service with Apache HTTP server, Microsoft Internet Information Service, or IBM WebSphere Application Server. The mobile devices and sensing terminals interact with the service application for context update, context query, and context-aware service consumption. The context data and users operations are stored in the database for further use.
- Interoperable data communication - The sensor data may be of different data type and data structure, thus the data interaction between devices is expected to be efficient and interoperable. The context data and commands are packed with universal techniques such as XML or JSON (JavaScript Object Notation) which are widely supported by the computing platforms.

The service end performs not only the context-aware service provisioning, but also the rule generation mechanism for high-level supervision of computing service customisation.

3.2.3 Fundamentals and Theories

In order to customise the computing service to the contextual situation, the gathering, reasoning, and distribution of context data are expected to be efficient enough. In addition, the context-aware service end needs to be able to create the corresponding context-aware service with learning capability which can handle the dynamic context intelligently. Therefore, appropriate models and theories are required in the efficient context data handling and effective context-aware service provisioning.

(1) Context-aware computing system model

In order to establish a context-aware system architecture, the context-aware system model is introduced to explicitly chart the relationship between user requests, contextual environment, and context-aware applications. The functional model for context-aware computing systems can be used to guide the design and implementation of the context-aware computing architecture. The context-aware computing system model and the context-aware system architecture are presented in Chapter 4.

(2) Context modelling and reasoning

In order to provide context-aware service customised to context of the user tasks, the context should be normalised with a context model for context distribution, computation, and service provision. The relevant context of the mobile platform is discussed and classified according to its source and characteristics. The context modelling approaches are further discussed with respect to the case of mobile-based context-aware environment, and the context model design and implementation in this investigation is presented in Chapter 5.

The context-aware service may be created based on some high-level context which may not be able to be obtained by the sensors directly. The context reasoning method is used to derive the high-level context from the low-level context or raw sensor data. The method for context reasoning is introduced in Chapter 6.

(3) Context-aware service customisation method

In context-aware systems, the computing service needs to be customised to the contextual situation with the context-aware service customisation strategy. The method should be able to deal with the heterogeneity of context and provide the appropriate computing service. Rule based service customisation which utilises the rule matching algorithm for the context-aware service provisioning is a promising solution.

(4) Rule generation mechanism for high-level supervision

The system should be able to create new adaptation rules or revise the existing rules for service provisioning using machine learning approaches. The mathematical methods such as BN, FL, kNN, and SVM can be used to discover the potential rules from a decision matrix of context and user operation. The rules extracted from the existing data are then used to supervise the context-aware service provisioning. With the rule generation algorithm, the context-aware service strategy can intelligently determine the underlying rules and provide proactive service accordingly. The method for rule generation using the history context is discussed in Chapter 6.

Within the design science research framework, this investigation is carried out with the above mentioned devices, techniques, platforms, models, and theories. Evaluation is conducted and conclusions are drawn based on the prototype system implementation and case studies.

3.3 Summary

This chapter firstly clarifies the aim and objectives of this investigation, and then gives an introduction and feasibility discussion of the method employed to carry out the research. The investigation follows the DSR methodology towards objective and generalised contribution. In what follows, Chapter 4, 5, and 6 are the development steps, Chapter 8 is the implementation and evaluation step, and Chapter 9 is the conclusion step.

Chapter 4 Context-aware Mobile Computing System Architecture Design

Through literature review, it is not difficult to find that lack of general system architecture is recognised as a practical problem in the field of context-aware mobile computing. Thus, it is significant to establish a general and flexible context-aware mobile system architecture to reduce the efforts and cost in context-aware mobile application development. In this section, a functional model is introduced to describe the relationship between users, contextual environment, and computer application in context-aware systems. A hierarchical system architecture with explicit defined functional modules is then proposed based on the context-aware system model for context handling and context-aware service provisioning.

4.1 Challenges and Requirements

Generally, context-aware mobile computing systems provide users with a computing service that is customised to a dynamic changing contextual environment in order to enhance the user experience of the applications. The heterogeneity in the devices and techniques, complexity of the physical environment, privacy and security issues of context data, and the high-level supervision capability of the system are identified as critical challenges, which have jointly resulted in the difficulty of context-aware system development.

In the context-aware system design, some problems have already been found and contributions have been made in the previous related investigations. Early research by Dey et al. (1999) and Grønli (2012) argued that the separation of sensing mechanisms and logic implementing can cause a looser binding between code and hardware, which could result in fast updating and easy maintenance. This point of discussion is accepted by most of the researchers and the layered architecture has become a general standard in context-aware mobile systems design (Schmohl and Baumgarten, 2008a; Zhang et al., 2009). For the complexity of environment, researchers suggest the possibility of integration of multiple devices, which may be conceptually similar to multi-sensor fusion techniques. Some applications with multiple wearable sensors are practical implementations of these design concepts. In addition, researchers have also reported the challenges in resource limitation of the mobile devices, and privacy issues in accessing users' private information as context data in the context-aware system design (Chen, 2003).

The influential factors that should be considered are not limited to the points mentioned above. However, the main work to do in the system architecture design is twofold: (1)

Efficient and interoperable acquisition, reasoning, and distribution of context data; (2) Context-aware computing service customisation strategy and high-level supervision mechanism. In order to reduce the efforts and cost in context-aware application development, the context-aware system architecture is expected to be general, flexible, and reusable, which is convenient for the building up, extension, and maintenance of context-aware applications. Therefore, the key requirements of the context-aware mobile system architecture determined are listed as follows:

- Be able to provide flexible interface for context acquisition from heterogeneous sources, such as physical sensors, user interaction, Internet data, etc.,
- Be able to represent real world objects, phenomena, values, and human preferences and actions in the context-aware computing system,
- Interoperable representation of context data and efficient distribution of context between the modules of the system architecture,
- Loose coupling between layers and functional modules for easy-maintenance,
- Flexible and convenient context API for context-aware mobile application design,
- Effective context-based service customisation strategy,
- Effective high-level supervision mechanism for proactive automation service, and
- Be able to guarantee the security and privacy of users' sensitive data.

Based on these principal issues, an open and general context-aware mobile system architecture is proposed in this chapter with the layered design pattern specifying the functionality of each layer and module.

4.2 A General Context-aware Mobile Computing Architecture

4.2.1 Functional Model of Context-aware Computing Systems

A significant strength of the context-aware system compared with the traditional ones is that, it can automatically gather the context information as implicit input to customise the computing service. Thus, the essential function of the context-aware mobile system is its capability to understand users' purpose, and proactively make a decision to provide the appropriate service accordingly. Schmidt (2002) proposes an iHCI model to describe the interaction between the ubiquitous computing system and the physical world.

To distinguish context-aware computing systems from traditional ones, the interaction models of the traditional computing system and the context-aware one are described in Figure 4-1 and Figure 4-2.

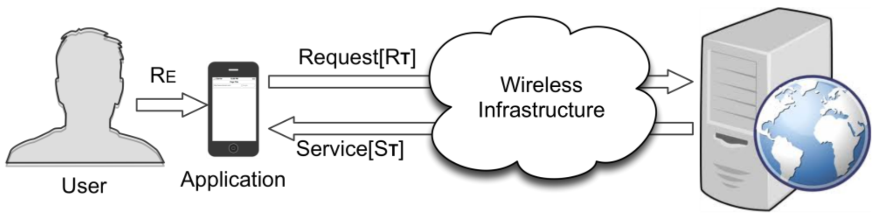


Figure 4-1 Traditional Mobile Computing Service

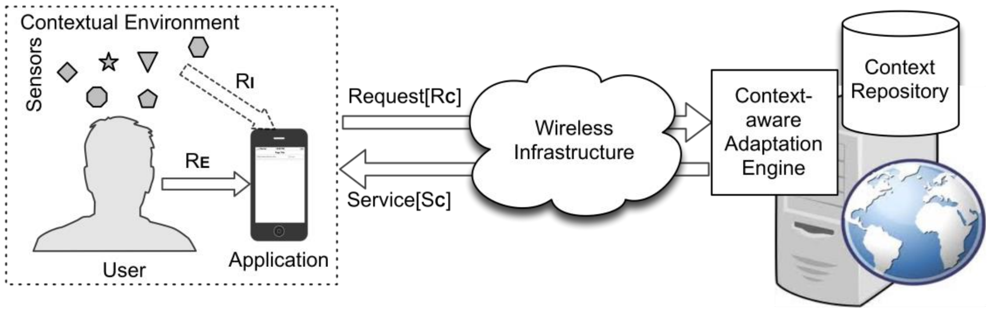


Figure 4-2 Context-aware Mobile Computing Service

In Figure 4-1, the traditional computing system provides the computing service - S_T according to user's explicit request - R_E, which can be obtained by the user's operation of the keyboard, mouse, or touch screen.

$$S_T = F(R_T) = F(R_E) \tag{4-1}$$

In contrast, as shown in Figure 4-2, the context-aware computing system provides the context-aware computing service S_C according to user's explicit request R_E and computing context information C related to the computing tasks.

$$S_C = F(R_C) = F(R_E, C) \tag{4-2}$$

where $C=\{C_u,C_d,C_n,C_e,C_h,\ldots\}$, and $C_u,C_d,C_n,C_e,C_h,\ldots$ are the contextual environment of computing tasks such as user context, device context, network context, environment context, and history context. With the implicit input C, the computing service S_C is customised to the contextual situation. Therefore, the operational efficiency and user experience is promoted if the context information C is obtained with the sensors automatically.

The context-aware computing paradigm is a user-centric design approach, which strives to reduce human effort while providing users with the appropriate computing service. The acquisition, delivery, management, and computation of context data increase the complexity of the system. However, it simplifies the complicated user operation and enables some novel proactive and automatic services with a variety of sensing techniques. From the user's perspective, the strengths of the context-aware systems are:

- It reduces the user's effort or supervision in the applications,
- Computing service is provided automatically or even intelligently,
- The complex system and technique employed are invisible to users, and
- The involvement of novel sensing techniques creates new functions.

Therefore, in the development of a context-aware computing system, the context information plays a role that is as important as the user's explicit input. The sensed context is some underlying information about the user which can be explored to characterise the user's requests, such as location, ambient environment, devices used, and user preferences. Admittedly, the handling of the context information is a critical task in the design and implementation of context-aware systems. Thus, a Hierarchical Context-Aware Computing (HCAC) architecture for context gathering, context management, context distribution and service adaptation is proposed towards a holistic solution for context-aware system development.

4.2.2 HCAC Architecture – Modules and Components

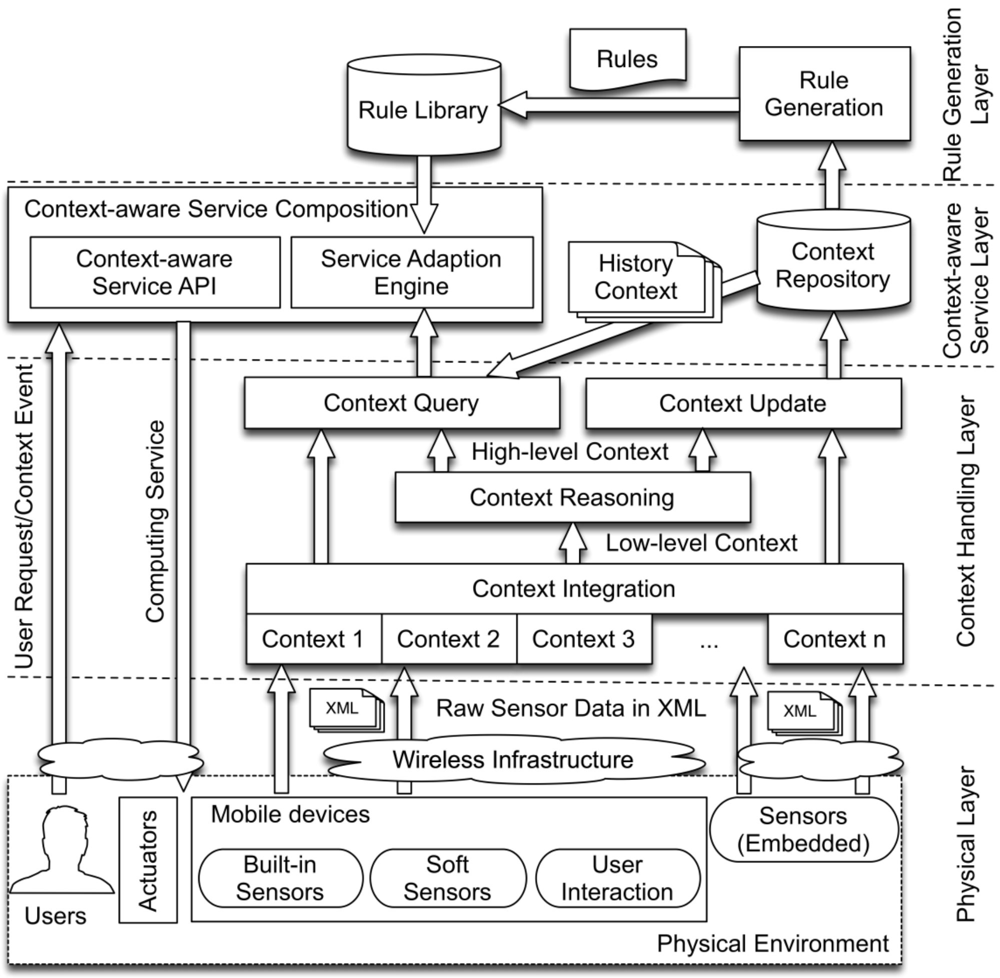


Figure 4-3 HCAC Architecture

In contrast to the system that is based on specific middleware technologies such as DPWS (Parra et al., 2009; Chen and Zhi, 2011) and OSGi (Gu et al., 2004; Lin et al., 2008), this investigation presents a generalised concept of context-aware system architecture which can be implemented with different techniques. As shown in Figure 4-3, the proposed HCAC architecture gives a clear view of the acquisition, distribution, and utilisation of context data.

The system is hierarchical since the four layers can independently fulfil integrated functions with technologies of different subject domain in a loose coupling way. In addition, the context data in the layers exists in the form of raw sensor data, structured data, formalised context, and domain knowledge respectively, which is hierarchical in semantic level. The physical layer and context handling layer perceive variables from physical environment and formalise the data to be structured context. Then, context-aware service layer uses related context to produce context-aware information service. The context data is then stored in the context repository as history context for the rule generation layer to discover new rules to supervise the context-aware service provisioning.

In what follows is a brief introduction of each part of the system within the four layers:

(1) Physical Layer (PL)

The physical layer is the interface between the physical world and the computing system. It is mainly comprised of the low-level hardware electronics observing the context and executing the adapted commands. Entities in this layer are human, environment, sensors, and actuators:

- Human: the service requester and the person to whom the service responded
- Environment: the ambient information of computation tasks
- Sensors: components used to collect the information relevant to the computing tasks
- Actuators: components used to execute the commands automatically

In this layer, some embedded electronics may be required to do some pre-processing, convert the logic for communication, or pack the data to particular formats. Devices like Analogue to Digital Converter (ADC), MCU, and wireless transceivers are needed to convert the physical variable, collect sensor data, and transmit data respectively.

(2) Context Handling Layer (CHL)

The context handling layer is responsible for the effective acquisition and sharing of context data. This layer behaves as an intermediate layer which connects the physical layer with the context service layer. This layer may include functional modules of: context integration, context update, and context distribution.

- *Context Integration* - This module gathers the context data and normalises the data for context service adaptation. Some simple pre-processing may be performed in the acquisition, such as lightweight filtering.
- *Context Update* - This module updates the context data in real-time once the context variable changes. In order to guarantee the real-time performance of the system, the update strategy considering the signal frequency and computation load plays an important role.
- *Context Reasoning* – This module obtains the high-level context with the low-level context or raw sensor data. It makes the context data more explicit for the computing system to understand and to use, thus makes the computing service less sophisticated.
- *Context Query* - The context query module enables the sharing of context data among the other modules in the system architecture. The efficiency of the context query may affect the efficiency of the context adaptation to a large extent.

Due to the heterogeneity of the techniques and devices used in the physical layer for acquisition and communication, efficiency and interoperability of the modules in this layer may largely affect the performance of the system.

(3) Context-aware Service Layer (CaSL)

The context-aware service layer achieves the main function of the composition of service customised to the dynamic context. It makes use of the context data in either the service end or the mobile client end, and provides the service with implicit requests. Thus, it reduces the user interaction and increases the degree of automation of the applications. The functional modules in this layer are:

- *Context-aware APIs* - The context-aware APIs provide interfaces for context-aware mobile applications. There are application interfaces for context data acquisition and context consuming. They interact with the mobile devices for user requests and service response.
- *Service Adaptation Engine* - The service adaptation engine is responsible for the composition of the service according to the user’s requests and the relevant context data. This module accesses the rules and creates the relevant service with the context data and then sends the results back to the users or executive components.

- *Context Repository* - The context repository stores the context data in the database system. The current context and the history data is stored in the context repository organised with context models for the system to use.
- *Client End Application* - The client end application, most likely a mobile client application, provides an interface for the user to interact with the computing system. It gathers the user's interaction and sensor data, then interacts with the service end and provides the user with the required information adapted to the contextual situation.

This layer handles the management and use of context at the service end. It utilises the context for context-aware service provisioning towards an adapted computing service.

(4) Rule Generation Layer (RGL)

The rule generation layer consists of modules gaining high-level supervision for applications. It employs the rule generation techniques to extract new rules or dynamically revise the parameters of the existing rules for service adaptation. These rules created automatically are used for supervision of the service adaptation.

- *Rule Library* - The context-aware computing service is created or composed with rules defined in advance, which are stored in the rule library. Of course, the rule library will be updated when new rules are created or existing rules are revised according to the context data and corresponding computing services.
- *Rule Generation Module* - This module derives new context adaptation rules from the history context and relevant computing service. The system can also adjust the adaptation rules by changing the parameters in order to better adapt to a user's contextual situation. The rule generation and adjustment tasks are done at service end.

This layer enables the system to be able to derive high-level rules to better supervise the service adaptation. Thus, it becomes smart with the learning capability of the system.

The strength of the layered HCAC architecture is its loose coupling structure between the functional modules. This system architecture is convenient to extend and maintain because of the explicit relationship of modules in the structure. The explicit structure also simplifies the implementation of context management. This investigation strives to make full use of context information to enhance the usability and efficiency of the mobile device-based applications. Thus, the four layers and the components for context acquisition, context handling, and context-aware adaptation are principal to this research. The relevant fundamentals, theories, and methods in the above mentioned four layers of the system architecture are discussed in detail in Chapter 5 and Chapter 6.

4.3 Establish the System in the Wireless Infrastructure

With the context-aware system architecture introduced in section 4.2, the four layers explicitly clarify the functionalities of the system in interacting with physical environment, context handling, service provisioning, and knowledge discovery. Then in this section, the method of building up the context-aware system with the electronic devices and techniques in the physical world is introduced, mainly including the communication interfaces, protocols and tools, and network topology which are appropriate for the case of context-aware systems.

4.3.1 Communication Interfaces between the Devices

The context information exists in the context-aware mobile computing system in different forms, as is shown in Figure 4-4. Originally, the context information is a physical phenomenon of the user or ambient environment, and it is converted into physical variables with explicit meaning when it is gathered by the sensing techniques. Then, it is changed into wireless signals in waveforms with the Wireless Access Point (WAP) for data distribution among the computing units. Finally, it is context data that can be used by the computing system. It can also be stored in different forms such as in database or in files.

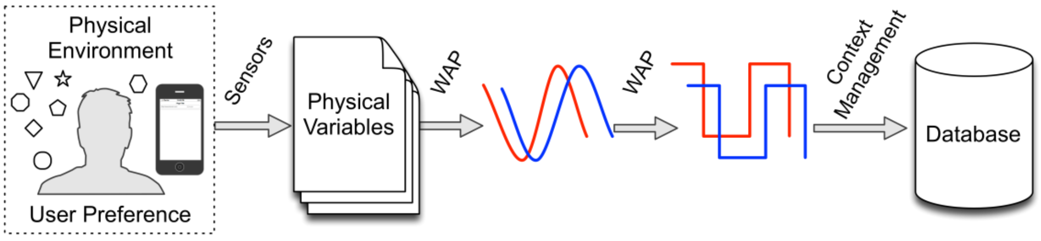


Figure 4-4 Format Conversion of the Context Information

The conversion of the context information from one form to another is achieved by processing units and communication interfaces to deliver the context data between them. The network connection in this investigation is based on the existing communication standards and infrastructure, which can be employed and established conveniently. For instance, if the WiFi is employed as the main infrastructure for communication, the sensors and actuators can be easily interfaced to a service end with WiFi access points such as WiFi transceivers and WiFi routers. Therefore, WiFi is selected as the main communication standard. From the communication perspective, the route of the context data can be described as the diagram given in Figure 4-5.

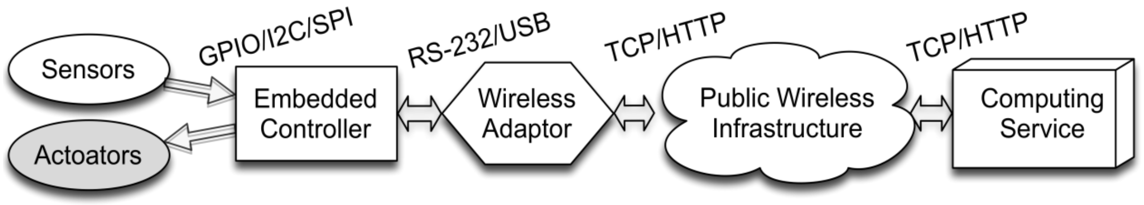


Figure 4-5 Communication Route of Context Data

(1) Embedded controllers/processors

The embedded electronics behave as controlling units in the physical layer of the architecture, which is directly placed in specified locations observing the physical environment. It manages data acquisition and data delivery from sensors to the communication interfaces which are connected to the WAP. It can be high performance CPUs such as ARM and DSP for computing intensive cases, or 8-bit MCUs for low frequency signal acquisition.

(2) General communication interfaces

In order to send the data to the WAP that allows the data to be available in a pervasive environment, the embedded controller needs to transmit the data to the WAP via a user-defined interface or universal interface, such as USB, RS-232, SPI, I2C, or GPIO. The bandwidth of context signal to transmit determines which serial port to use.

(3) Wireless transceivers

The wireless transceivers integrate the separate electronic devices into the wireless network. They receive the context data through the general communication interfaces and convert the data into a wireless form, such as WiFi, Bluetooth, 3G, or 4G. They also receive and convert wireless signals to control the actuators through the general communication interfaces.

(4) Wireless infrastructure

The wireless infrastructure consists of the wireless nodes, WAPs, and public wireless network that can bring the context-aware system into practical implementation. It breaks the constraints of the location and enables the whole system to be a service “anytime” and “anywhere”. As WiFi is widely employed in home and office areas, the cost of building up the context-aware system is largely reduced compared to the strategy of setting up a new network covering the target areas.

4.3.2 Protocols and Standards

For the efficient distribution of context data between the devices and components, the methods used are listed as follows for different purposes. These methods should be as general and interoperable as possible so that they can be supported by the devices and platforms.

(1) HTTP – context-aware service invocation

HTTP is the most commonly used protocol in client-server distributed systems for information retrieval. In this system, HTTP is the main protocol for communication between the client (mobile client and wireless sensor node) and server in the wireless network. Since HTTP is widely supported by mobile platforms and wireless nodes, the main tasks such as user request, context update, context query, and service provision can be conveniently achieved.

(2) TCP/UDP – P2P connection

For the devices within the same physical domain, the devices can be connected with a TCP/UDP socket for P2P communication. Different from stateless HTTP, the TCP is a stateful protocol. So, the entities can request voluntary data transmission with the TCP connection. It is expected for some use cases when clients in the context-aware system may need to talk to each other.

(3) XML/JSON

The data transmission between the wireless node, client, and server requires a lightweight and efficient data representation. The XML and JSON are competent candidates for the lightweight and interoperable requirements for this case. Therefore, XML schema is employed for data representation and parsing between the devices in the distributed system.

The goal of employing the protocols and standards is to guarantee the interoperability and efficiency distribution of the context data. The context distribution network is built with the above communication interfaces, protocols, and standards to fulfil the context-aware service adaptation. In order to promote the performance, the communication interfaces, protocols, and data format may differ for different use scenarios.

4.3.3 Network Topology

Practically, there may be plenty of devices wirelessly connected in the context-aware computing system. In order to address the devices in the network and distribute the context data efficiently, the connection needs to follow some strategies. The widely used wireless network topologies can be categorised into infrastructure-based network, Ad-Hoc network, and hybrid network, which differ in organisation of the devices in the wireless network.

(1) Infrastructure-based network connection

The infrastructure-based connection is the traditional fixed network system which is commonly used in the client server systems. The specific topology of wireless infrastructure may be bus, star, ring, mesh, and tree, etc. The infrastructure-based topology ensures high data availability, but there are threats of single point-of-failure in centralised network components.

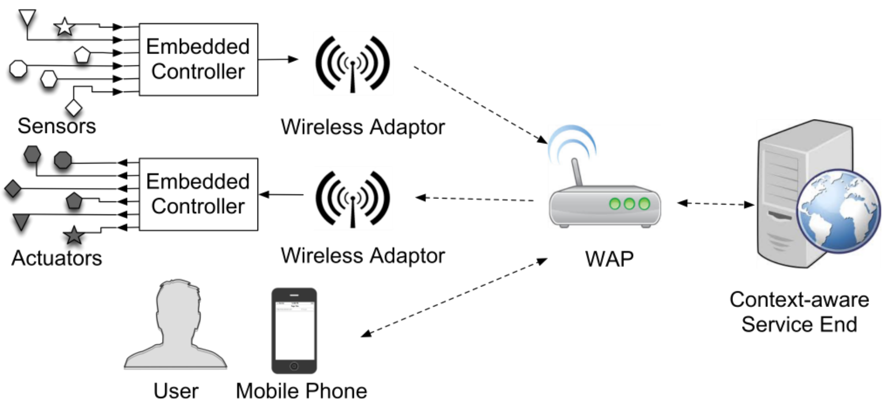


Figure 4-6 Infrastructure-based Network Connection

(2) Ad-Hoc network connection

The Ad-Hoc connection is a way that is prospective for scenarios where the wireless infrastructure is not available, such as Wireless Sensor Network (WSN). As the Ad-Hoc is implemented in a decentralised way, it supports context transmission among the different mobile nodes. The shortcomings of this method are its difficulty in management and weakness in data availability and problem handling.

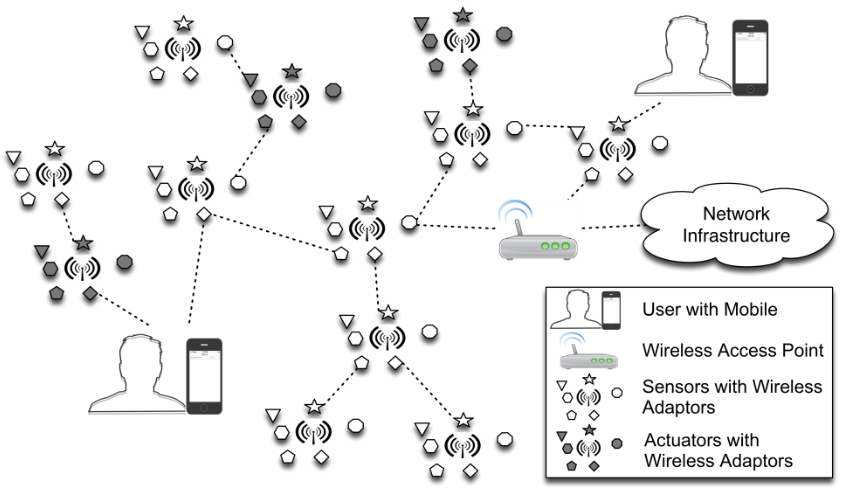

Network Infrastructure
User with Mobile
Wireless Access Point
Sensors with Wireless Adaptors
Actuators with Wireless Adaptors

Figure 4-7 Ad-Hoc Network Connection

(3) Hybrid network connection

The hybrid network connection combines the strengths of both the fixed and Ad-Hoc ones. The fixed part can guarantee the high data availability of the system, and the Ad-Hoc part may reduce the overheads of the infrastructure and enable a necessary link which is not available with only fixed parts.

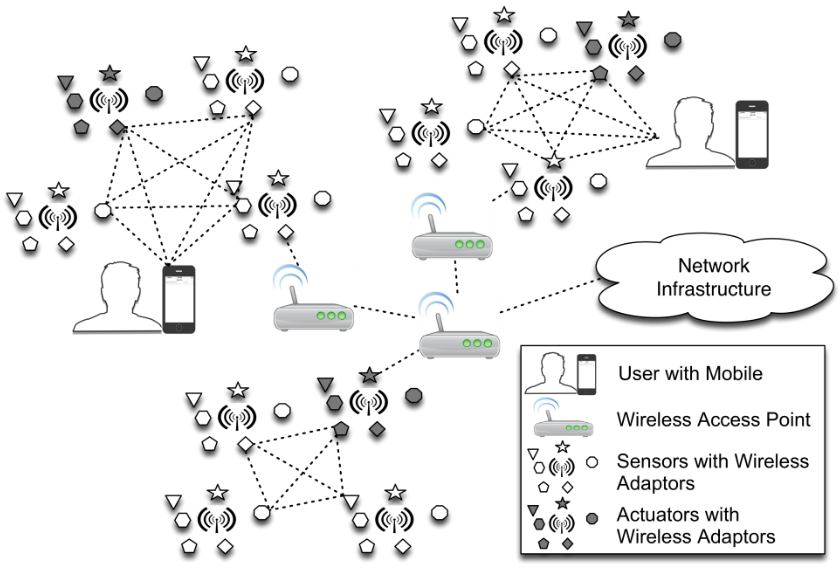

Network Infrastructure
User with Mobile
Wireless Access Point
Sensors with Wireless Adaptors
Actuators with Wireless Adaptors

Figure 4-8 Hybrid Network Connection

In the context-aware systems, all the above network topology may be feasible; which one to employ is determined by the complexity of the physical contextual environment. In the case studies, the infrastructure-based architecture is selected to guarantee the stability and data availability. In addition, easy reachability is another reason why a WiFi infrastructure-based network is selected.

4.4 Service End of the Context-aware System

It is a typical characteristic of the context-aware mobile service that the service requests are multiform, where different request may relate to different context items. For efficient handling of context as a significant resource, the RESTful client-server architectural style is employed for context-aware service application development. The explicit use of HTTP and XML, JSON transfer are principles of the design paradigm, which result in simplicity, flexibility, lightweight, and low coupling between the interacting entities. Therefore, the RESTful APIs are suitable to be used dealing with context update and context distribution. The flexibility of this design pattern in context-aware service application development leverages the notion of “context as a resource”.

4.4.1 Context Update and Context Query

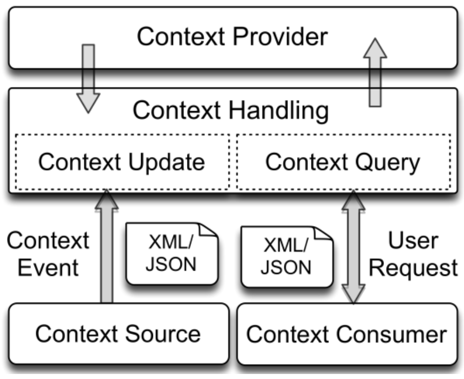


Figure 4-9 Entities and Process of Context Distribution

In the context-aware system architecture, the main entities are: sensors, context-aware computing server, users and actuators. From the context utilisation perspective, these entities can be considered as context source, context provider, and context consumer. The entities and basic context delivery structure of context-aware systems can be described with Figure 4-9.

In the context-aware service provisioning, the computing tasks are distributed to the context sensor nodes, mobile clients, and service end. The context data needs to be up to date for timely service adaptation in the dynamic environment. As the context data varies frequently, the context value needs to be gathered from the context source and saved to the context repository in real-time. According to the modularisation design principle, specific interfaces for context update are designed as follows:

- *Function environmentContextupdate (ec1, ec2, …, ecn);*
- *Function userContextupdate (uc1, uc2, …, ucn);*
- …
- *Function socialContextupdate (sc1, sc2, …, scn);*

In the above functions, *ec1, ec2, …, ecn, uc1, uc2, …, ucn*, and *sc1, sc2, …, scn* are items of environment context, user context, and social context. With the context data that is relevant to the computing task user requests, the context handling layer can then compose the appropriate computing service accordingly. Similarly, the functional interfaces handling users' requests are specified as follows:

- *Function tablelightControl(command_Light_i);*
- *Function tablefanControl(command_Fan_j);*
- …
- *Function restaurantRecommendation(user_Id);*

Obviously, much of the computation is done at the context-aware service end since all the work is for the realisation of context-aware adaptation. The explicit use of HTTP methods make the service end application simple and clear, and the XML/JSON transfer of context in the client server design pattern enhances interoperability between the heterogeneous devices.

4.4.2 Event-driven Context-aware Service Model

In contrast to some client-server interactions in many service architectures, the context-aware server occasionally needs to actively notify the clients with some messages. When an event occurs, the server will initiate new communications with the specific clients. The event-driven communication can also minimise the coupling between sensors, actuators, and components in service applications. The events in the context-aware mobile systems may be classified into two categories:

(1) *Context update* – When context data varies or exceed some threshold, the relevant event will happen according to the service customisation rules;
(2) *User request* – When user makes requests, the corresponding service will be invoked according to the context condition.

Implementation of the event-based service can be based on the polling, long polling, and WebSocket. By polling, the client queries the state change at service end continuously. Thus, the client can monitor the state change at server end. By long polling, the server keeps an open connection and transmits an event description when an event occurs. The WebSocket is a TCP based full-duplex protocol which makes more interaction possible between client and server, facilitating live content interaction. In the case studies, long polling with HTTP connection is employed in the client service architecture. This method is easy to implement at the service end with the commonly used technologies.

4.4.3 Context-aware Service with High-level Supervision

The context-aware service provisioning can be regarded as a decision making procedure, where the condition attributes are the context input and the decision attributes are context-aware services.

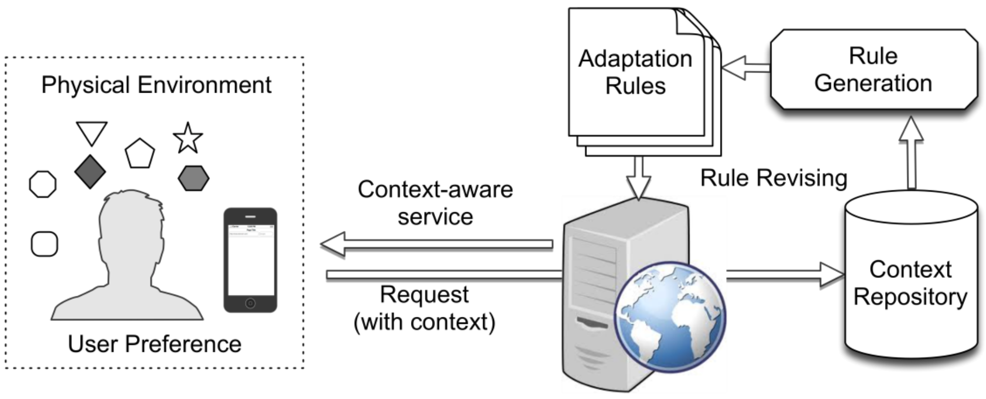


Figure 4-10 Context-aware Adaptation with High-level Supervision

The rule generation layer is one of the highlights of this research which distinguishes it from most of the previous works. It entitles the system with the capability of learning to customise the service to be more appropriate to the contextual situation. The traditional pure rule-based system can be considered as an open-loop structure in cybernetics, while the architecture with the context rule generation layer can be considered as a closed-loop structure. The open-loop structure provides a context-aware service with just context adaptation rules. In this way, the rules may not be suitable to new use scenarios when some of the context environment changes. But the closed-loop structure in this investigation utilises the context and users' feedback to generate new rules supervising the service adaptation. Thus, the system will learn to adjust the adaptation rules to suit the new use scenarios automatically without human supervision.

The context-aware service provisioning strategy and rule generation mechanism are presented in Chapter 6.

4.5 Summary

This chapter firstly clarifies the challenges and requirements of context-aware system architecture design, and then proposes a HCAC architecture with a functional model. The layered functional modules in the architecture constitute a loose coupling structure, which makes the system convenient to extend and maintain. Thereafter, the communication infrastructure, network topology, protocols and standards, and context-aware service end that can be employed to establish the system architecture are also briefly introduced. In what follows, the methods and techniques in the context acquisition and modelling will be elaborated in Chapter 5, and the fundamentals and theories of the context service layer and rule generation layer will be elaborated in Chapter 6.

Chapter 5 Context Data Modelling, Acquisition, and Representation

To guarantee the performance of the context-aware systems based on the hierarchical architecture proposed in Chapter 4, two essential targets must be achieved: (1) efficient and interoperable distribution of context; and (2) easily understanding and flexible utilization of context data by applications. Thus, this chapter firstly provides some insight into the characteristics of context data for smartphone-based use scenarios, and then elaborates context modelling, acquisition, and data representation methods for interoperable distribution, easy understanding, and flexible utilisation of context.

5.1 Context based on Mobile Built-in and Embedded Sensing Techniques

In mobile and pervasive environments, the context information is states or variables describing the ambient environment or characterising users and computing tasks. Researchers argue that integration of the built-in sensors is the future prospect for context-aware service, which is one of the key drivers of mobile applications (Lane et al., 2010). The embedded sensor and Radio Frequency Identification (RFID) techniques have invaded into many fields of people's daily life. The heterogeneous devices and techniques therefore raise challenges in gathering, normalising, representing, and managing context data in an efficient and interoperable way. For this reason, this section will provide some insights into the context and characterise the features of various context data, in order to determine appropriate methods for the handling and processing of context data.

5.1.1 Features of the Context information

Henricksen (2003) has identified the characteristics of context information in the context-aware pervasive computing environment: heterogeneity, uncertainty, and dependencies. In addition, the technical advances in mobile devices and various sensing techniques have enriched the content of context. Therefore, context data with novel sensing techniques are equipped with new features, which are summarised as follows:

- *Heterogeneity* - The context data are of different types, precision, and semantic meanings, thus they are difficult to use for context-aware adaptation.
- *Uncertainty* - The acquisition of context data is prone to noise and errors due to various reasons such as environmental interference and resolution of sensor.

- *Diversity* - The context of computing tasks is not limited to ambient environment and computation power, but extended to high-level context such as human behaviour. Some can be obtained directly with sensors, and some can only be obtained by reasoning.
- *Novel technique involvement* - Context is from a variety of different sources and some of them can be obtained using novel techniques which were previously unavailable, such as RFID tags recognition and barcode scanning on mobile.
- *Computation intensive task available* - Context acquisition methods may be based on lightweight gathering or computation-intensive algorithms on the resource limited terminals, such as voice recognition.
- *Environment complexity* - Context acquisition and context-aware service requests may happen in complex environment, such as on the move, roaming between wireless networks, and in noisy environments.

The above characteristics of context information have increased the difficulty in context data acquisition, normalisation, representation, and understanding for context-aware service adaptation. There have been heuristic investigations into mobile-oriented context acquisition and handling systems, such as SenSay (Siewiorek et al., 2003), mobile sensing platform (MSP) (Choudhury et al., 2008), and MobiSense (Waluyo et al., 2010). But most of the investigations are application-specific, which are used for usability of phone, health monitoring, information recommendation, etc. In this investigation, with the identified characteristics of the smartphone context in pervasive environments, the context information is classified for the sake of efficient acquisition and utilisation of context.

5.1.2 Classification of the Context Information

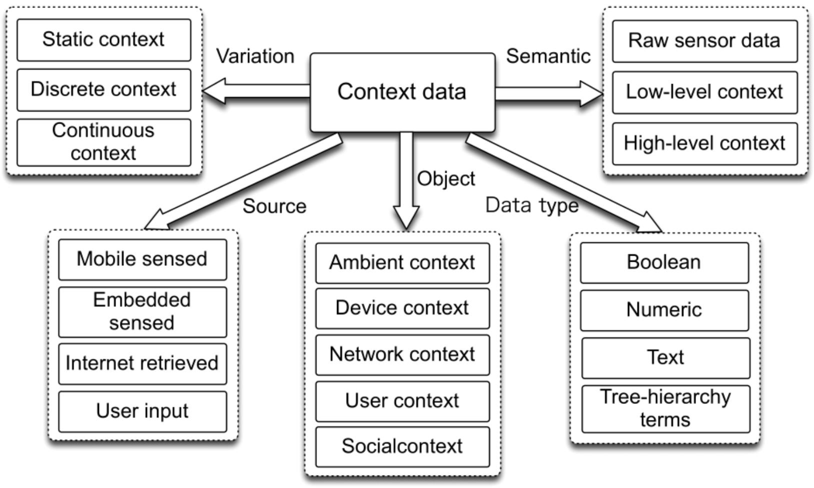


Figure 5-1 Classification of Context Information

In order to obtain the context information in an organised way, and then represent it in a rational format for efficient semantic understanding, the context information is divided into different categories. The purpose of the classification is to pursue the common ground for in-depth understanding of context, and find methods of handling different kinds of context data. Context items of the same class may have some general features that can be handled with similar methods in acquisition, representation, and computation. Then, as is shown in Figure 5-1, the context information can be classified in the following ways:

- *Variation frequency* - Signal of context variables may vary in different forms, the context data can be assorted into static context, continuous context, and discrete context. The static context may be constant for certain circumstances, such as the smartphone screen size. The continuous context varies with the time, such as the temperature and brightness. Then, the discrete context is usually Boolean state variables, such as human body sensor output.
- *Semantic level* - The context data are data that is used to describe the real world. Thus, they are of different semantic levels in the application domain, which can be classified into raw sensor data, low-level context, and high-level context. Take temperature for example, "-10 ℃" is raw sensor data, "Cold" is low-level context, and "Central heating on provided user is in" is high-level context.
- *Source of context* - From the context source respective, the context can be assorted into: (1) mobile sensed context - the data obtained with the mobile built-in sensors; (2) embedded sensed context - the data obtained with embedded sensors; (3) user input context - which collects users' interaction as context, and (4) internet retrieved context - data accessed from Internet sources, such as weather information.
- *Object context describes* - From objects the context data describes, it can be sorted into environment context, device context, network context, user context, and social context. This classification is clear and easy to understand, so is thus widely accepted in context modelling.
- *Data type of context* - Context data may be of different data types and structures, such as Boolean, Numeric, Text, Tree-hierarchy terms, etc. The different data types require different methods for the computation of context data in service provisioning.

In addition to the above classification, the context data can be assorted into participatory sensed context and opportunistic sensed context according to the involvement of people as sensing device custodians. Then, in people-centric sensing systems, context can be assorted by the sensing methods: personal sensing, social sensing, and public sensing (Khan et al., 2012). The various classification methods have demonstrated the heterogeneity of context data and implied the common ground of different context data for the exploration of shared methods in context acquisition, normalisation, and representation.

Due to the imperfection feature of context data, it is an essential task to incorporate the QoC parameters into the computation in the process of context acquisition, distribution, and context-aware service customisation. According to Table 1-1, the main QoC indicators - up-to-dateness, correctness, precision, resolution, and trustworthiness are considered as the most important QoC indicators describing the context data in this investigation. Some QoS descriptions such as timestamp and sensor resolution are integrated with context data as the meta-information for context-aware service adaptation. The context provider and context handler in the system architecture contain functional modules to deal with QoC. The context provider provides the variables relevant to the QoC parameters, and the context handler then evaluates the QoC with these variables. The representation of context data with awareness of the QoC is discussed in section 5.3.2.

5.1.3 Requirements and Challenges

The purpose of the discussion and analysis of context information is to pursue appropriate methods for context acquisition, modelling, and presentation towards efficient context-aware service customisation. Although the computing tasks are user-centred, divergence exists in the different application domains. These application-specific requirements that characterise a programming model for the development of context-aware applications are: privacy, scalability, synchrony, extensibility, and reusability (Biegel, 2004), unobtrusive acquisition, and QoC indication.

The main challenges facing the context acquisition, modelling, representation, and storage are summarised as follows:

- How to collect the context data in an organised way considering divergence in varying frequency, data structure, data source, etc.?
- How to employ the mobile built-in sensors and arrange the sensing techniques in an unobtrusive way with minimised human supervision?
- How to pre-process and normalise the diverse context data to be convenient for computation and management?

- How to represent the heterogeneous context data in a standard format that is machine understandable and interoperable between computing terminals?
- How to handle the context with an extensible and expressive context model for ubiquitous use for different application domains?
- How to integrate the QoC to describe the data quality and supervise the context-aware service customisation accordingly?
- How to store the context data, user operations, and context-aware service that can be potentially used for rule generation to derive the high-level rules supervising the context-aware service customisation?

The above aspects are the essential challenges identified in this investigation facing the context data acquisition and utilisation in context-aware systems, which are the main tasks to solve for this chapter. Therefore, the rest of this chapter strives to explore the methods of modelling, gathering, pre-processing, and representing the context data.

5.2 Context Modelling

Normally, in context-aware systems, there is a significant gap between the sensor data and the level of information that is useful to the applications (Hoareau and Saton, 2009). The purpose of context modelling is to bridge the gap by using a formal description of the concepts and relationship of context information to make it usable for the context-aware systems. With a formal description, the context of different data types and from various sources can be integrated, reasoned about, and used for high-level computation. A formal context model can promote the sharing and exchange of context information between applications, and provide information representations that are appropriate for the applications to process (Sahafipour and Javidan, 2012). For the mobile-based context-aware systems, context data is distributed between context sources and context consumers involving a variety of heterogeneous techniques and devices. Therefore, an appropriate context model which can describe the context data systematically and logically is essential to the implementation of context-aware systems.

5.2.1 Context Modelling for Mobile and Pervasive Use Cases

In the smartphone-based pervasive systems, the smartphone is a handheld device which is appropriate to be used for characterising users' behaviour and ambient environment. It also plays an important role as an information interface for user interaction between user and context-aware computing system. The main use scenarios of context-aware mobile systems based on the built-in and embedded sensing technique can be classified into the following categories:

- Enhancing the human mobile interaction to reduce human efforts in using the mobile applications restricted by data input and information representation, such as web content tailoring and image content-based searching with built-in camera.
- Obtaining users' specific context with mobile built-in sensing techniques to provide context-aware service in real-time, such as GPS navigation, tour guiding, and NFC for security or payment.
- Automatic and intelligent control of ambient facilities in smart space with the distributed sensors and actuators based on embedded electronics.
- Using the mobile device as a user interface to access context-aware services, such as news, music, and movie recommendation.

To implement the above use scenarios, the relevant context information should be provided to accomplish the context-aware computing tasks. It raises requirements for the context model to be able to handle service adaption, resource awareness, and user interface adaptation, etc. (Preuveneers et al., 2004). For context-aware computing systems, the update, query, reasoning, and utilisation of context data are all based on operation of context data normalised with context model structures. The context model organises the context data logically towards interoperable distribution and efficient computation of context data. The role context model plays can be described with the diagram in Figure 5-2.

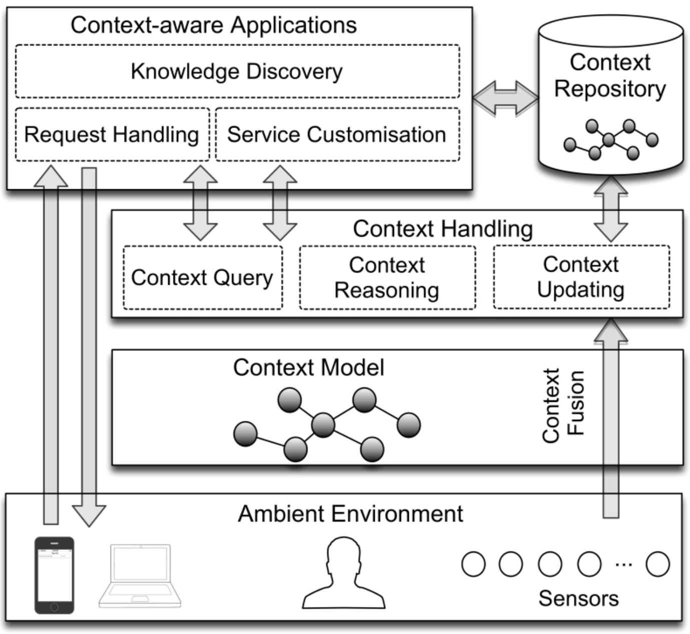


Figure 5-2 Context Model in Context-aware Computing Architecture

The context model transforms the variable describing the physical world to the form of high-level data model that supports the delivery and usage of the right information when necessary. It is found that the context model affects the efficiency of context updating, context reasoning, and service adaptation. Thus, the context modelling should be based on the in-depth analysis of attributes and the inter-relationship of context data, and its flexibility in context representation and context distribution. In this section, the context modelling approach is presented in order to conduct efficient collaboration of the low-layer context acquisition, middle-layer context management, and high-layer service provisioning.

5.2.2 Context Modelling

Discussion in Chapter 2 favours ontology-based context modelling in several context modelling approaches. Compared with the other alternatives, it is strong in representing concepts in the complicated environment. It can provide formal semantics for context knowledge about the objects, relationships, and domain constraints, which support the sharing and integration of structured context information (Poveda-Villalon et al., 2010). Hence, it is a competitive candidate to express meanings and relations of the context information in pervasive environments. Therefore, it is selected for context-aware mobile computing in this investigation.

Obviously, it is impractical to have a global model which can describe all concepts that may be included in a context-aware service (Huang et al., 2005). For most cases, a user request or computing task of some applications may not be related to so many context items. However, a holistic context model that incorporates the relevant context items for different application domains, which is extensible for general use covering most use scenarios, is practically required. In this investigation, the context is categorised and modelled considering both the generality of the context model and the broad application domains. To build the context model, the main entities in the context-aware systems are determined as follows:

- *User* - the entity that requests the service, and to whom the information or computing service is provided.
- *Device* - the entity that is used as a hardware platform to request and receive the context-aware service or collect context data.
- *Network* - the entity that builds the wireless and pervasive environment for the context distribution and service provisioning.

- *Environment* - the ambient conditions that affect the computing service which can be characterised to enhance the computing tasks.
- *Service provider* - the service end that handles the context information and provides context-aware computing service for the users requesting it.

In context-aware systems, the context can be regarded as the information that is used to characterise the relevant entities and objects that affect the computing service. To this end, the context is categorised into the following dimensions in this investigation, from the perspective of objects the context describes:

- *User Context (UC)*
  The user context consists of the user preference, user activity, user emotional state, and user social context that may affect the user’s expectation of the computing service.
- *Computation Context (CC)*
  The computation context includes information about the device hardware and software platforms that are used to request or execute the context-aware service.
- *Network Context (NC)*
  The network context means the attributes of the wireless connection for the mobile device that affects the data communication and service provisioning.
- *Environment Context (EC)*
  The environment context denotes the ambient environment in which the computing task is executed. They are normally some physical variables characterising the ambient situation of the user and computation task, such as temperature, brightness, noise level, humidity, barometric pressure, etc.
- *Location Context (LC)*
  The location context is about the position where the context-aware computing task is executed. It can be information about street, building, room number, or longitude-latitude pairs.
- *Time Context (TC)*
  The time context is about at what time the context-aware computing task is executed. It may be expressed with year/month/day, time of the day, day of week, etc.
- *History Context (HC)*
  The history context is the existing context information in the context repository, which can be used to extract some supervision rules to guide the current operation or execution of computing service.

Regarding the entities and context domains, the context model can be described with the diagram of centre context terms as shown in Figure 5-3.

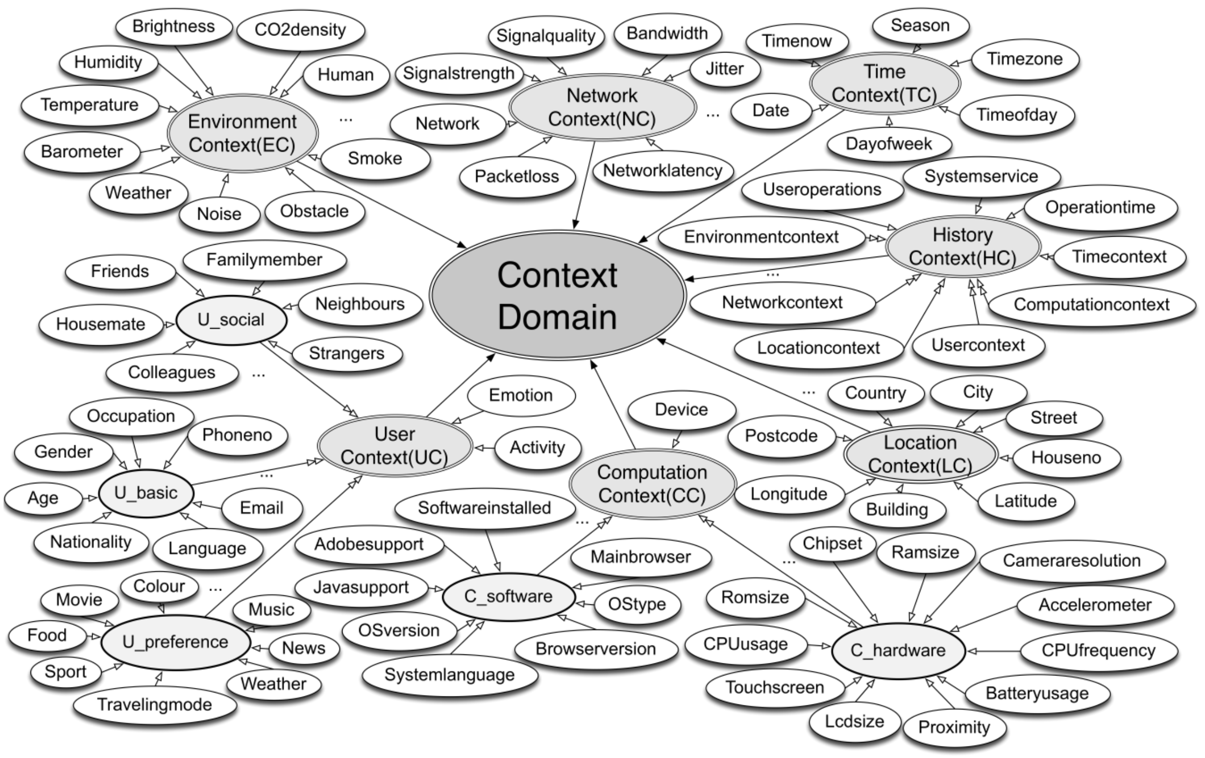


Figure 5-3 Graphical Representation of Context Model

In Figure 5-3, the above seven context classes constitute the high-level terms of the model, and context source modules such as physical sensors, soft sensors, and the user interaction modules that provide raw context data constitute the low-level terms. Each of the above classes is comprised of the sub-classes that are considered to be critical in the context-aware service. The relationship between the classes and sub-classes is built based on the concepts and facts that are practical in the real world. The purpose of context modelling is to formalise the context which describes the physical environment of computation tasks, and then enhance the sharing and semantic understanding of context by the computer systems. According to the graphical representation of the context terms, a vocabulary including the data types, properties, and interrelationships are built for the sharing and semantic understanding of context information. The centre context terms in Figure 5-3 can be tailored to fit the different use cases of different application domains.

The context data is collected and represented in a structured format for interoperable distribution between the devices and context-aware service customisation with the context model. In simple words, the strengths of this context modelling method are:

- *General* – it is generalised and not limited to any application specific domains.
- *Flexible* – it is convenient to be extended and tailored to different application domains.
- *Expressive* – it is strong in the capability of expressing possible application domains.
- *Explicit* – it defines explicit semantics between the entities for context computation.
- *Usable* – the simple structure can be easily represented for context distribution and then processed with mobile platforms and service applications.

With the context model described above, the context information can be normalised and shared between the computing devices for context-aware service provisioning. The following sections will then offer insight into the acquisition and representation of context data in a pervasive environment based on mobile and embedded electronics.

5.3 Context Acquisition and Context Data Representation

With the context model presented in section 5.2, the sensor data can be normalised and structured for the understanding and processing by context-aware applications. In this section, the prospective methods for context acquisition and context data representation are discussed.

5.3.1 Context Acquisition – From Raw Sensor Variable to Context Data

It is common sense that, the methods for signal acquisition and data processing may be determined to some extent by the source and attributes of context data. The context data is gathered from various sources with heterogeneous techniques as stated in section 5.1.2. Therefore, the context acquisition methods are introduced according to the source of context information: mobile built-in sensor acquisition; embedded sensor acquisition; accessing Internet source; user manual input. The methods are introduced with relevant examples.

(1) Mobile built-in sensor acquisition

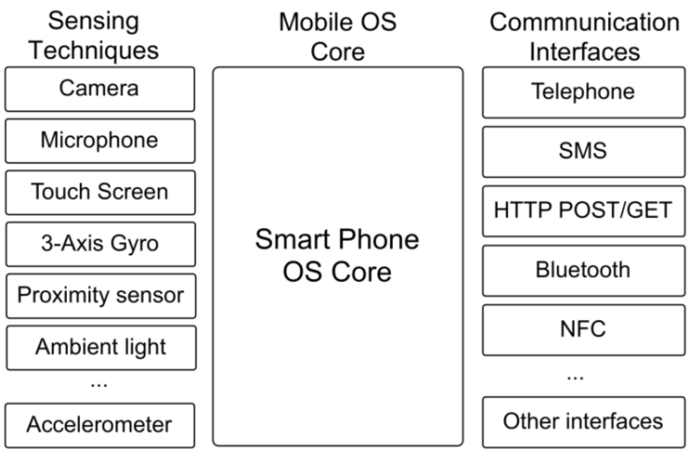


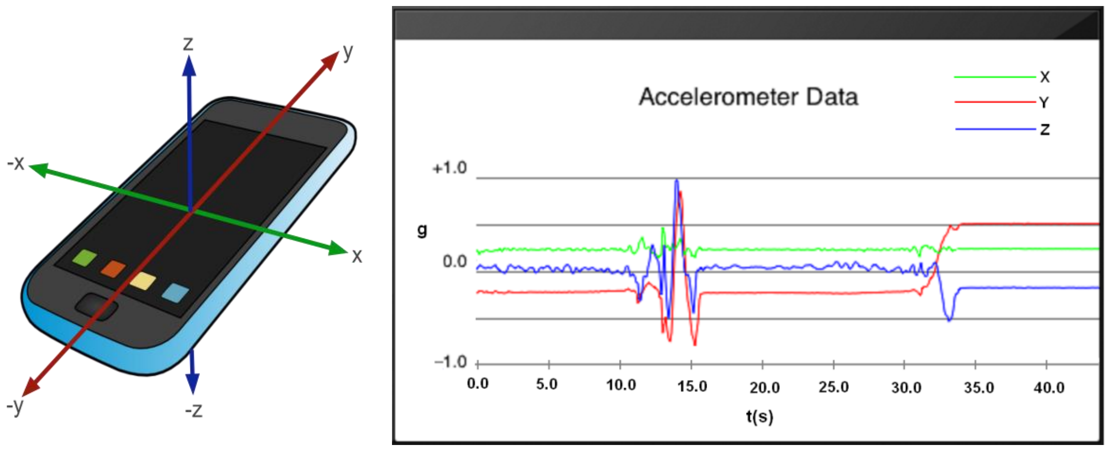
Figure 5-4 General Framework of Smartphone Mobile Platform

The acquisition of context with the smartphone built-in sensors is an essential method to characterise the user who requests the service. This kind of context can be accessed directly by the sensor on the devices with relevant mobile application, either lightweight or computation intensive. For the current smartphones, the commonly available sensors are enumerated in section 2.3.2. Figure 5-4 gives a general framework of the smartphone mobile platform consists of the mobile OS core, sensing techniques, and communication interfaces.

Two smartphone-based context acquisition techniques which will be used in case study 1 in Chapter 8 are introduced below. One is human activity detection with built-in accelerometer, and the other is barcode scanning with built-in camera.

Example 1: Human activity detection – A built-in accelerometer-based lightweight approach

The 3-axis accelerometer is a kind of sensor which is gaining widespread attention in human activity recognition and even further human behaviour modelling and analysis. The fundamentals of the 3-axis accelerometer are as shown in Figure 5-5. The three axes x, y, z in Figure 5-5 (a) are the three orthogonal directions, and the variables in red, green, and blue colours are the acceleration in the three directions. When the object device moves, the acceleration data in the three directions can be obtained by the accelerometer. The sample data for user activity detection in case study 1 Chapter 8 is as shown in Figure 5-5 (b).



(a) (b)

Figure 5-5 3-axis Accelerometer and Sample Data

With the 3-axis accelerometer, simple activities such as shaking or rotating the phone can be detected directly with the mobile applications. The sample code for iOS platform shaking detection is as follows:

```
NSString * const shakedNotification = @"shaked";
UIAccelerometer *accel = [UIAccelerometer
      sharedAccelerometer];
- (void)accelerometer:(UIAccelerometer *)accelerometer
      didAccelerate:(UIAcceleration *)acceleration
{
   if (fabsf(acceleration.x) > kAccelerationThreshold  ||
      fabsf(acceleration.y) > kAccelerationThreshold  ||
      fabsf(acceleration.z) > kAccelerationThreshold )
   {
      if(cameraOn == false){
          cameraOn = true;
          NSNotificationCenter *sk = [NSNotificationCenter
      defaultCenter];
          [sk postNotificationName: shakedNotification
      object:self];
      }else{
          cameraOn = false;
          [self dismissModalViewControllerAnimated:YES];
      }
   }
}
```

List 5-1 Shake Detection with 3-axis Accelerometer for iOS Platform

This example is a use case in Chapter 8 case study 1. When users shake the smartphone, the application will be notified to open the camera for barcode scanning if the camera is off, and it will dismiss the camera view if the camera is already on.

For complex activities such as running, walking, and jumping, the detection is much more difficult. Some machine learning techniques such as BN, SVM, kNN, FL, and ANN may need to be involved to improve the accuracy in activity recognition. This field has attracted much research interest in the recent years for health, fitness, elderly care, etc.

Example 2: Barcode scanning – A built-in camera-based computation intensive approach

Barcode scanning is already used in some recent mobile applications. In these applications, 1-D barcode and 2-D QR code are used to encode data for convenient access by Internet-enabled mobile devices. In this example, a barcode format "Telepen" of UK university student ID card scanning application is implemented, which is used in an EU Commission funded project 'Mobile Exam System (MES)'. In this project, the students used mobile devices for classroom examination. It was required that the students could login to the system by student ID card with a barcode on the surface.

The information of all the 1-D barcode formats is encoded and represented by the wide and narrow bars and spaces, and it is the same with Telepen format. The barcodes of the Telepen format consist of only four patterns of bar and space pairs. These four patterns of Telepen format are given in Figure 5-6, and an example Telepen barcode is given in Figure 5-7.

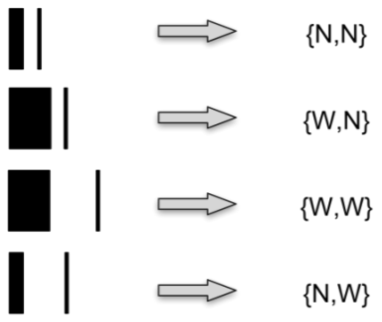


Figure 5-6 Patterns of Telepen Barcode Format

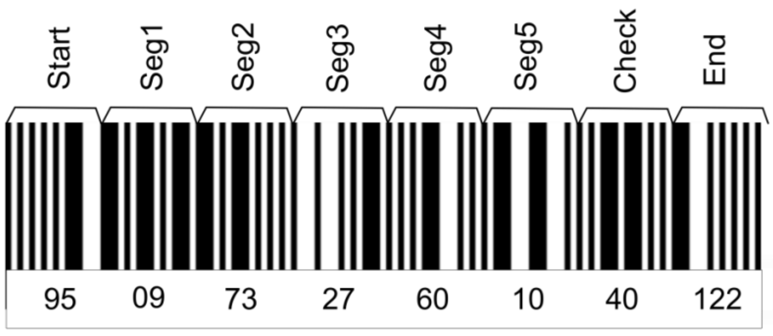


Figure 5-7 A Sample Barcode of Telepen Format

As shown in Figure 5-7, the first and the last segments are the start and end of the whole barcode respectively. Besides the start and end, Seg1 to Seg5 are the data segments and segment 6 is the verification segment. Each segment in the barcode is encoded with the above mentioned four patterns: {N, N}, {W, N}, {W, W}, and {N, W}, where N means narrow and W means wide.

For mobile devices, the barcode can be obtained with the built-in camera, and the decoding computation can be achieved with image processing algorithms. The essential task of the barcode scanning module is the decoding algorithm which extracts the encoded data from the camera gathered images. It consists of the format recognition, pattern matching, 8-bit parity checking, result verification, and length checking. However, besides the decoding algorithm, the system needs to do some preparation work such as camera initialisation, camera lens focusing, image acquisition, etc. The flow chart of the barcode scanning algorithm is given in

Figure 5-8. In the decoding algorithms, the data normalisation, pattern matching algorithm, and special consideration for Telepen format are the major concerns.

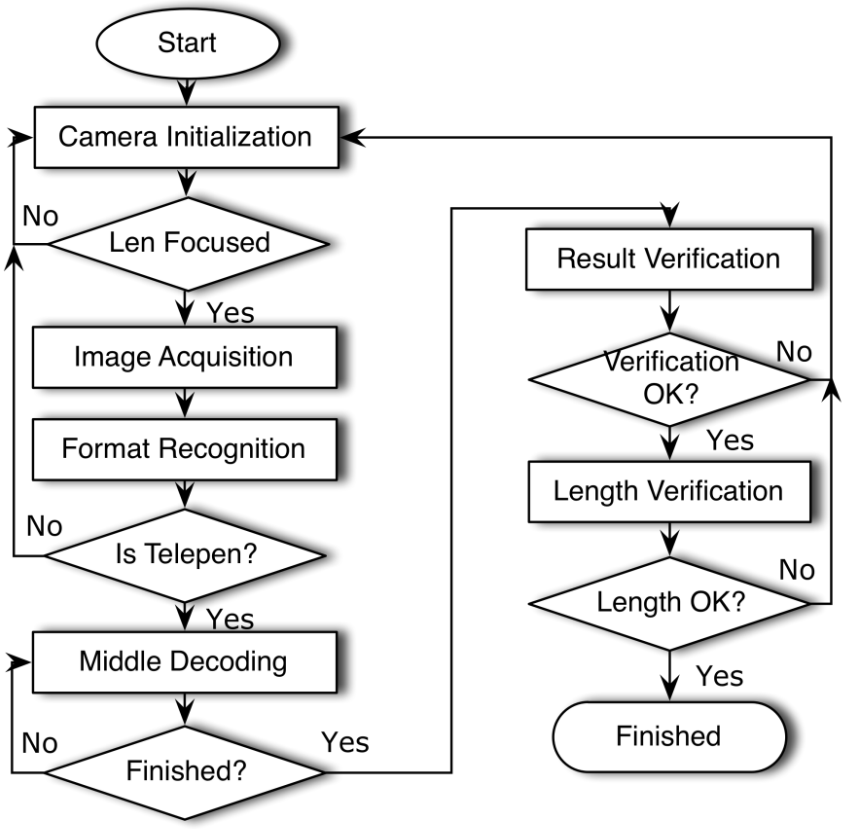


Figure 5-8 Flow Chart of Barcode Scanning with Mobile Built-in Camera

The Telepen barcode decoding algorithm is created with a barcode scanning framework ZXing library (ZXing Group, 2013), in which the Telepen format was not included in the supported decoding formats. The decoding algorithm is developed in objective C, Java, C#, and C++ for different mobile platforms. Finally, the barcode scanning model is integrated into mobile client applications of the MES project on mobile platforms iOS, Android, WP7, and Symbian^3.

(2) Embedded sensor acquisition

Besides the built-in sensors on smartphones, the context can also be obtained with sensors that can be placed in the ambient environment with embedded electronics. For instance, the temperature sensor and smoke sensor can be used to monitor the temperature and smoke of the ambient environment. The acquisition of this kind of context may need the sensors, embedded controller, and wireless nodes. Three examples about human body sensor, obstacle avoidance sensor, and temperature sensor used in case study 2 in Chapter 8 are elaborated as follows.

Example 3: Human body sensor

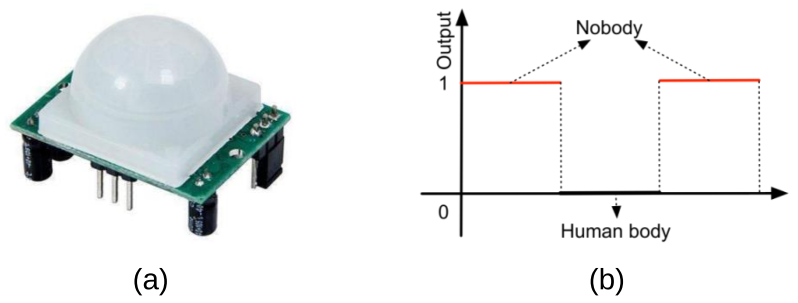


Figure 5-9 Human Body Sensor and Its Output

For the human body sensor, the human body can be detected when someone appears in the sensing area, and the output turns to '1' from '0' as indicated in Figure 5-9. Sensor output can be obtained using MCU GPIO port with the same voltage standard.

Example 4: Obstacle avoidance sensor

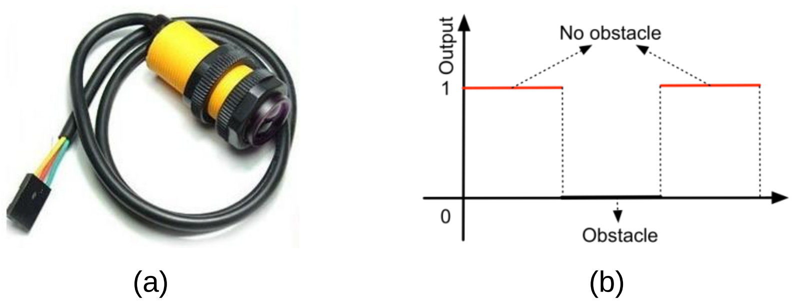


Figure 5-10 Obstacle Avoidance Sensor and Its Output

The obstacle avoidance sensor and its signal output are as shown in Figure 5-10. It can detect the objects within its sensing range. When an object is sensed, the output turns from '1' to '0'. The use of the obstacle avoidance sensor is similar to that of the human body sensor.

Example 5: Temperature sensor

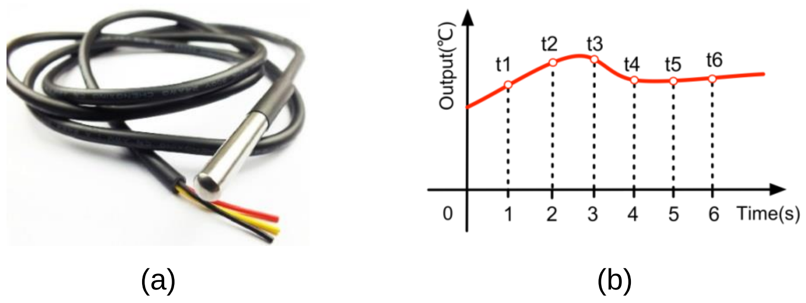


Figure 5-11 Temperature Sensor DS18B20 and Its Output

The temperature sensor can be used to measure the temperature of the ambient environment. It provides discrete acquisition of continual signal rather than binary '0' and '1'. The output interface of the sensor is a unique 1-Wire data bus.

(3) Accessing Internet sources

There is some context information that can be accessed from specific providers, such as weather forecast and IP positioning. For this kind of context, the mobile devices simply access the data from specific addresses and parse the obtained data to get the required values.

Example 6: Weather information from third-party API

For the weather information, API by the service provider ‘worldweatheronline’ can be used. The mobile can access the weather data in different data formats with the post code, city name or IP address. The mobile device can send request by HTTP POST with the command ‘http://api.worldweatheronline.com/free/v1/weather.ashx?q=Huddersfield&format=json&num_of_days=5&key=hr8rwdcru9d93tvvh2ddpc8t’, and the weather information can be accessed as shown in List 5-2.

```
{
  "data": {
    "current_condition": [{
      "cloudcover": "50",
      "humidity": "58",
      "observation_time": "01:41 PM",
      "precipMM": "0.2",
      "pressure": "1006",
      "temp_C": "10",
      "temp_F": "50",
      "visibility": "10",
      "weatherCode": "116",
      "weatherDesc": [{
        "value": "Partly Cloudy"
      }],
      "weatherIconUrl": [{
        "value":
"http:\/\/www.worldweatheronline.com\/ima
ges\/wsymbols01_png_64\/wsymbol_0002_s
unny_intervals.png"
      }],
      "winddir16Point": "W",
      "winddirDegree": "260",
      "windspeedKmph": "48",
      "windspeedMiles": "30"
    }],
    "request": [{
      "query": "Huddersfield, United
Kingdom",
      "type": "City"
    }],
    "weather": [{
      "date": "2013-04-18",
      "precipMM": "2.9",
      "tempMaxC": "11",
      "tempMaxF": "51",
      "tempMinC": "3",
      "tempMinF": "37",
      "weatherCode": "353",
      "weatherDesc": [{
        "value": "Light rain shower"
      }],
      "weatherIconUrl": [{
        …
      }],
      "…
    },
    …, {
      "date": "2013-04-22",
       …
      "weatherDesc": [{
        …
      }],
      "weatherIconUrl": [{
        …
      }],
      …
    }]
  }
}
```

List 5-2 Weather Context from Internet in JSON Format

Then, with JSON parsing framework, the variables can be conveniently obtained and utilised by context-aware applications.

Example 6: IP positioning

Similar to the weather information access, the positioning of the user can be obtained from the public service provider. The freegeoip.net provides a free API for mobile devices to get the position information with IP address in the format: http://freegeoip.net/{format}/{IPAddress}. The format of accessed data can be XML, CSV, and JSON. Using the author's IP address, the request string can be: http://freegeoip.net/xml/161.112.232.221. The corresponding XML document retrieved from the service provider is given in List 5-3.

```
<Response>
   <Ip>161.112.232.221</Ip>
   <CountryCode>GB</CountryCode>
   <CountryName>United Kingdom</CountryName>
   <RegionCode>G8</RegionCode>
   <RegionName>Kirklees</RegionName>
   <City>Huddersfield</City>
   <ZipCode/>
   <Latitude>53.650001525878906</Latitude>
   <Longitude>-1.7833000421524048</Longitude>
   <MetroCode/>
   <AreaCode/>
</Response>
```

List 5-3 IP Positioning Information in XML Format

Therefore, the position information can be easily obtained by the context-aware applications with a XML parser.

(4) User manual input

Besides the above methods, some applications provide user interfaces to collect a user's manual input that can be used as context information. Most of the context information obtained with this method may not change frequently. For example, some sites keep the user's contact and preference as context data with the user's permission, and the information automatically appears in required operations. And also in some applications, some threshold values or event-based settings may be required as context for automatic execution of some tasks. The strength of this acquisition method is that the context information gathered is explicit; the weakness is that the acquisition of context is cumbersome, as the manual input on mobile devices may be time consuming and error prone.

5.3.2 Representation of Context towards Interoperable Distribution

The resource limitation of the mobile devices requires the representation of the context data to be flexible, efficient, and lightweight, and heterogeneous environment requires the context description to be interoperable and easily understandable.

5.3.2.1 Tools for Context Data Representation

A data description language appropriate for the representation of the most fundamental assertions involving data should allow at least the ability to mark names of properties and types as such (Ram, 2004). There are some general frameworks available for representation of the context data in context-aware pervasive environment, such as web ontology language and resource description framework. There are also frameworks built based on the general frameworks which can be tailored for some use cases, such as Composite Capability/Preference Profiles (CC/PP) and User Agent Profile (UAProf) specification. These frameworks can be used to deliver device context and guide the customisation of context-aware applications. However, the context in a pervasive mobile system refers not only to the device capability but also the more plentiful context of other application domains. Thus, for flexibility and efficiency, this investigation employs the interoperable techniques for the representation of context data for context distribution and utilisation.

For the current mobile devices, two data formats are widely accepted: XML and JSON. The composing and parsing tools are supported by all the mainstream mobile platforms, such as iOS SDK, Android SDK, Windows Phone SDK, etc. A brief introduction to these two tools is given as follows:

(a) XML

The W3C interoperable document-oriented means XML, which is used for describing tree-based structured information, is an ideal tool for context data description. This format is supported by most of the available development platforms, and it is the mainstream data-interchange standard for the network-based data communications. The main strengths of XML format are:

- Human and machine readable,
- Strong representation ability,

- Open and extensible,
- Strict syntax, simple and efficient parsing algorithm,
- Platform and language independent, and
- Widely adopted by computer industry.

(b) JSON

The JSON is a lightweight data-oriented data interchange format for data communication between network connected devices. It is a language independent text format that is very easy to generate and parse for the machines, and it is also not difficult for human to read and write. Thus, the JSON text format is also a competent data interchange language for wireless devices in mobile and pervasive environment. Likewise, the strengths of JSON format are:

- More lightweight and simple in structure,
- Open and extensible,
- Concise format due to name-value pair-based approach,
- Simple API for different languages,
- Platform and language independent, and
- New but increasingly accepted.

Apparently, there are much in common between XML and JSON. Both of them are competent for most use cases. For a sample tree-based structure of context fragment as shown in Figure 5-12, the relevant XML and JSON description is given in List 5-4 and List 5-5.

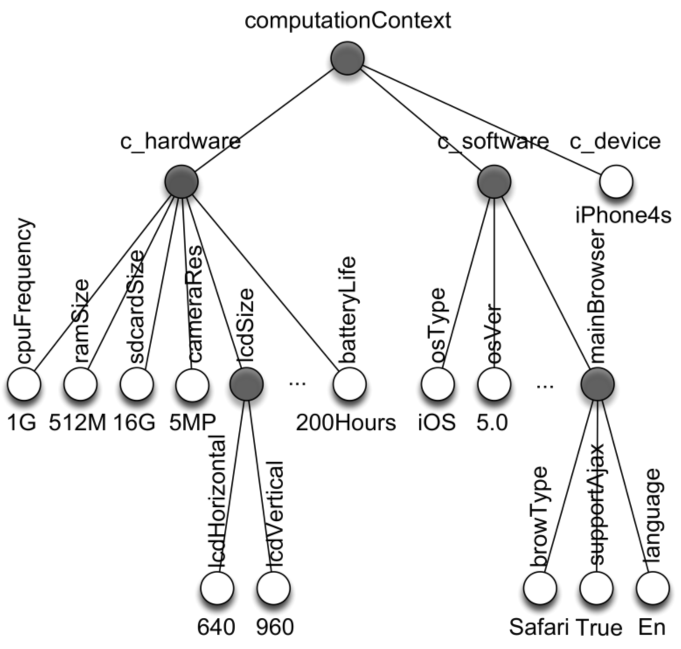


Figure 5-12 A Sample Context Fragment

```
<?xml version="1.0" encoding="UTF-8"?>
</ComputationContext>
  <c_hardware>
   <cpuFrequency>1GHZ</cpuFrequency>
   <ramSize>512M</ramSize>
   <sdcardsize>16G</ sdcardsize>
   <cameraRes>5MP</cameraRes>
   <lcdSize>
       <lcdHorizontal>640</lcdHorizontal>
       <lcdVertical>960</lcdVertical>
  </ lcdSize >
   …
   <batterylife>200Hours</batteryLife>
   </c_hardware>
   <c_software>
     <osType>iOS</osType >
     <osVer>5.0</osVer>
     …
     <mainBrowser>Safari</mainBrowser>
        <supportAjax>TRUE</supportAjax>
        <language>En</language>
      </mainBrowser>
   </c_software>
   <c_device>iPhone4s</c_device>
  </computationContext>
```

List 5-4 XML Representation of Context Fragment

```
{
   "computationContext":{
     "c_hardware ":{
          "cpuFrequency":"1GHZ ",
          "ramSize":"512M",
          "sdcardSize":"32G ",
          "cameraRes":"5MP ",
          "lcdSize": {
              "lcdHorizontal":"640",
              "lcdVertical":"960"
          },
          …,
          "batteryLife":"200Hours"
     }
   },
   {
          "c_software": {
               "osType ":"iOS",
               "osVer":"5.0",
               …,
               "mainBrowser": {
                   "browserType":"Safari",
                   "supportAjax":"TRUE",
                   "language":"En"
               }
          },
        {
          "c_device": "iPhone4s"
          }
        }
    }
```

List 5-5 JSON Representation of Context Fragment

Since XML and JSON each have their strengths, they are both employed in this research. XML is employed for some scenarios when the data is complex in structure, or the service provider can only provide XML documents. JSON is for some lightweight occasions such as data interchange between mobile platforms.

5.3.2.2 Data Structure for Context Information Representation

The context-aware applications usually adopt some implicit policies or restrictions to context access which directly influences the performance of context management (Rocha and Endler, 2005). The context data gathered is expected to include some description information indicating the sensor information, validation period, precision, etc. The description can be some meta-information indicating the QoC to improve the quality of context-aware service customisation. For convenient management, the data structure of the context data is discussed for the sake of effective utilisation of the context data in the context-aware applications. The perceived context contains important characteristics that can be considered in the context data structure design as described by Gray and Salber (2001): (1) context values - the context properties that are gathered deliberately, and (2) metadata of context data – description of context including the QoS attributes, source of the content, etc.

To summarise, the attributes that are essential to the utilisation of context identified are listed as follows:

- *Context value(Cv) – the value of obtained context item*
- *Context acquisition method (Ca) – how the context data is obtained*
- *Context source(Cs) – device, sensor ID, or site the context data is from*
- *Data type (Dt) – the data type of the context value*
- *Context precision (Cp) – precision of context data*
- *Time of acquisition(freshness) (Ta)- date and time of context (YYYYMMDDHHMMSS)*
- *Time of validity (Tv) – the time of validity of context data*
- *Trustworthiness(Tw) – trustworthiness of the context data*
- *Reliability(Re) – reliability of the context data*

In the above QoC parameters, trustworthiness is used to describe context information regarding the attributes of “worthy of trust or confidence; reliable” (Simpson and Weiner, 1991). Reliability is the probability of failure-free acquisition of context in a specific environment. The format of the obtained context data is organised in the format as follows:

$$Con[contextitem] = [Cv][Ca][Cs][Dt][Cp][Ta][Tv][Tw][Re] \qquad (5\text{-}1)$$

With this structured representation format, two examples illustrated in section 5.3.1 are then represented with JSON format as follows:

Example 1: Barcode context

Con[barcode]=[0973276010][builtincamera][iphone4s][text][0.975][20130420220000] [Forever] [0.8][0.9]

The structured context data in JSON format for Example 1 is as shown in List 5-6.

```
{"contextBarcode":[{
        "Cv":"0973276",
            "metaInformation":[{
                "Ca":"builtincamera",
                "Cs":"iphone4s",
                "Dt":"text",
                "Cp":"0.975",
                "Ta":"20130420220000",
                "Tv":"Forever",
                "Tw":"0.8",
                "Re":"0.9"
            }]
    }]}
```

List 5-6 Structured Data for Barcode Context as an Example

Example 2: Weather context

*Con[weather]=[
{"current_condition":[{"cloudcover":"50","humidity":"58","observation_time":"01:4
1PM","precipMM":"0.2","pressure":"1006","temp_C":"10","temp_F":"50","visibilit
y":"10","weatherCode":"116","weatherDesc":[{"value":"PartlyCloudy"}],"weatherIc
onUrl":[{"value":"http:\/\/www.worldweatheronline.com\/images\/wsymbols01_png_
64\/wsymbol_0002_sunny_intervals.png"}],"winddir16Point":"W","winddirDegree":
"260","windspeedKmph":"48","windspeedMiles":"30"}]}]
[http://www.worldweatheronline.com][iphone4s][JSON][0.9][20130418134100][12hour
s][1][0.95]*

The structured context data in JSON format for Example 2 is as shown in List 5-7.

```
{"contextWeather":[{
        "Cv":[{
            "current_condition":[{
                "cloudcover":"50",
                "humidity":"58",
                "observation_time":"01:
                41PM",
                "precipMM":"0.2",
                "pressure":"1006",
                "temp_C":"10",
                "temp_F":"50",
                "visibility":"10",
                "weatherCode":"116",
                "weatherDesc":[{
                "value":"PartlyCloudy"
            }],
          "weatherIconUrl":[{
              "value":"http:\/\/www.w
              orldweatheronline.com\/i
              mages\/wsymbols01_png
              _64\/symbol_0002_sunn
              y_intervals.png"
            }],
            "winddir16Point":"W",
            "winddirDegree":"260",
            "windspeedKmph":"48",
            "windspeedMiles":"30"
          }]
        }],
        "metaInformation":[{
            "Ca":"http://www.worldweat
            heronline.com ",
            "Cs":"iphone4s",
            "Dt":"JSON",
            "Cp":"0.9",
            "Ta":"20130418134100",
            "Tv":"12hours",
            "Tw":"1",
            "Re":"0.95"
        }]
    }]
}
```

List 5-7 Structured Data for Weather Context as an Example

With the structured data described above, the context data is packed with the meta-information indicating QoC and the relevant attributes that are useful to the context-aware applications. However, some attributes describing the context may be very difficult to acquire, such as trustworthiness and reliability. After all, this context data representation format can enhance the understanding of context among the computing modules, which actually promotes the efficiency of context distribution and processing in context-aware systems.

5.4 Summary

This chapter goes into details of the context data to characterise its features, and the context modelling, acquisition, and representation methods are provided for context-aware service customisation. Through the analysis of the context that is available, the classification of context data is given regarding its main features. According to the classification, the context modelling method for mobile built-in modules and embedded sensing techniques based systems, and the four context acquisition methods are presented for various context data from different context sources. Finally, the context representation method is provided with XML and JSON towards lightweight machine-understandable structure. In succession, the methods for context reasoning, context-aware service customisation, and high-level supervision will be discussed in Chapter 6.

Chapter 6 Context Computation and Context-aware Service Customisation

The context model and data representation methods presented in Chapters 4 and 5 can normalise the context data and connect the physical world with the computing systems. Another important process for context-aware service customisation is to derive high-level context with the low-level context. In this chapter, two essential issues for context-aware service provisioning are discussed: (1) context integration and context reasoning to formalise the context data and infer high-level context; (2) context-aware service customisation strategy and rule generation mechanism. Evaluation of context-aware mobile application is then briefly discussed, considering its specificities.

6.1 Context Abstraction and Context Integration

In context-aware systems, context integration can be defined as *the process of combining data residing in different sources and providing the system with formalised data for context-aware service adaptation* (Lenzerini, 2002). As discussed in previous chapters, the context data is characterised by the attributes of being independent, distributed, heterogeneous, inconsistent, and error prone. The context source may produce large amounts of raw context data that should be semantically integrated and filtered based on the user's preference and the application context (Bolchini et al., 2006). Researchers find high-level and abstract context is more useful and important for sophisticated context-aware applications (Zimmer, 2006). Romero et al. (2010) argue that failure to integrate the diversity of context sources in a standard and flexible way is a common problem of existing context integration approaches.

This section provides methods for context abstraction and semantic integration to deal with the discrete context towards context-aware service adaptation. To handle the heterogeneous data from different context sources, the critical work is determined as follows:

(1) Context abstraction – to extract features from raw sensor data.
(2) Context integration – to combine context data from heterogeneous data sources and provide formalised data.

Through context abstraction and context integration, the raw sensor data is then processed and converted into formal context data which can be used to derive high-level context and for context-aware adaptation.

6.1.1 Context Abstraction

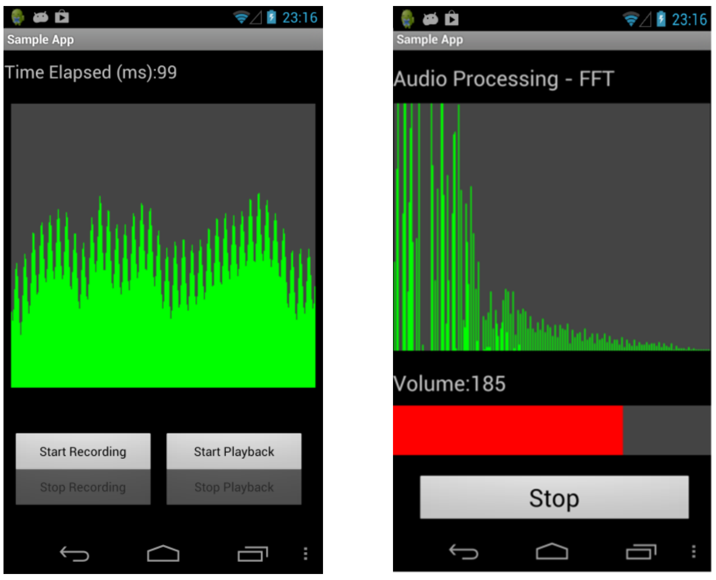
Normally, the raw sensor data may accompany some imperfection attributes, which make it unqualified to be used directly. In particular, some of the context data can be explicitly obtained with the sensing techniques. This kind of context data can be used directly for context updating, as the values of the context items are defined with particular meanings. For instance, the position information from the GPS module:

$$C\_gps = [Latitude, longitude]$$

The data pair denotes the latitude and longitude of the user. Then, for the human body sensor:

$$C\_human = \begin{cases} 1, & Nobody\ is\ in\ the\ sensed\ area \\ 0, & Somebody\ is\ in\ the\ sensed\ area \end{cases}$$

However, some context data can’t be directly gathered with the sensing techniques; they can only be abstracted by the computation of the raw context data. The tasks of obtaining the context data may be computation intensive, and most of the work can be done at the mobile client or embedded device end. Typical examples of this kind of context are noise level, barcode, and human behaviour which are hidden in microphone sound signal, images with camera, and 3-axis accelerometer data. The audio signal collected using the microphone on Android phone Google Nexus S is shown in Figure 6-1. Figure 6-1 (a) gives the waveform of the environment sound signal, and Figure 6-1 (b) provides the FFT spectrum and volume of the environment sound signal.



(a) (b)

Figure 6-1 Noise Level from Microphone Signal

The underlying features are hidden in the raw sensor data collected by the sensing techniques, and the methods used for feature extraction depend on the format of the sensor data. In this investigation, the threshold-based approach is the main method employed for context abstraction, which is suitable for the sensors employed in this work. But sometimes further computation may be required to extract the features in advance from raw sensor data. The basic threshold-based abstraction can be represented as follows:

$$f(x) = \begin{cases} 1, & x > Threshold \\ 0, & Otherwise \end{cases} \tag{6-1}$$

Noise level detection is a typical example that can be evaluated with the threshold-based approach. With the above function, the noise level *f(x)* can be noisy (1), or not noisy (0), where input *x* is the audio signal and threshold is some specific level of noise set in the system. When the input value is greater than the threshold, the environment is considered to be noisy; otherwise, it is not noisy. Of course, the parameter that is used to compare with the threshold may be based on calculations such as averaging.

In the same way, the value of temperature sensor and brightness sensor can be abstracted by using the threshold-based approach. The value of the brightness sensor between 0 and 65535 can be segmented as follows:

$$V_{Brightness} = \{Ultra\ low: <5;\ Low: 5\text{-}50;\ Medium: 50\text{-}1000;\ High: >1000\}$$

Similarly, for the temperature sensor:

$$V_{Temperature} = \{Ultra\ low: <0;\ Low: 0\text{-}15.9;\ Medium: 16\text{-}28;\ High: 28\text{-}35;\ Ultra\ high: >35\}$$

With the threshold-based approach, some numerical data can be interpreted with an abstract variable that is appropriate for computation in the context-aware framework.

However, for some use cases, the pattern matching approach may be needed for feature extraction, such as speech recognition, human behaviour recognition, and barcode scanning as mentioned above. The output signal of the sensor is usually time-sequence function such as sound and 3-axis accelerometer signals, or even sophisticated ones such as visual signals. The system can try to match the selected attributes of the raw sensor data with the sample patterns and calculate the similarity to find the optimal matching.

Through the context abstraction, the useful information in imprecise and vague raw context data is made explicit and appropriate for formal representation and context integration.

Obviously, context abstraction is an essential step to bridge the gap between the raw sensor data and abstract context values that can be used by the context-aware computing system.

6.1.2 Multiple Context Data Integration Framework

The data abstraction can be regarded as a pre-processing procedure of the raw sensor data. After that, the heterogeneous context data can be normalised with a formal structure that can be accepted and utilised by context-aware applications. The formatted data can be described with meta-information indicating the attributes of the context data. Normally, data integration is based on centralised system architecture for context-aware applications. The sensing, data collection, and feature extraction are implemented at the distributed mobile or embedded client's end, and the formalised structured context is stored in a centralised database system at the service end.

In context-aware systems, it is common that only part of the context data is relevant to specific context-aware computing tasks, and the context-aware computing tasks are highly dependent on the relevant context. In order to organise the context in an efficient way for context query, context reasoning, and context-based service provisioning, the entities in the context-aware environment and the context items describing them are explicitly defined in the system. The context integration framework is as shown in Figure 6-2.

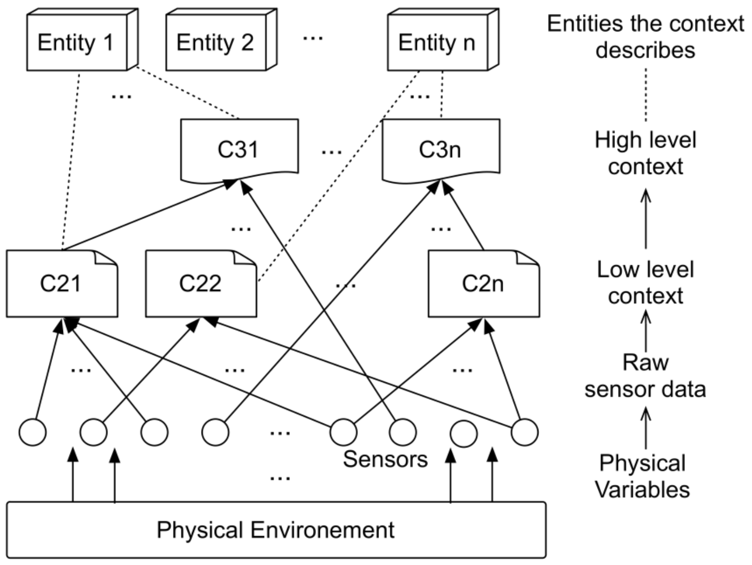


Figure 6-2 Context Integration Framework

In Figure 6-2, the hierarchical context integration framework consists of functions of data collection, feature extraction, and context inference to obtain the different levels of context data. The context inference here means the process of obtaining high-level context from low-level context data with context inference models.

As stated in Section 4.4, the client server design pattern is employed for context handling regarding its simplicity, efficiency, flexibility, and convenient development. The context-aware system architecture specifies the functional components for exchanging context information following the modularisation design principles. The combination of the design pattern and context-awareness leverages the notion of 'context as resource' to efficiently handle the multiple context data from heterogeneous sources.

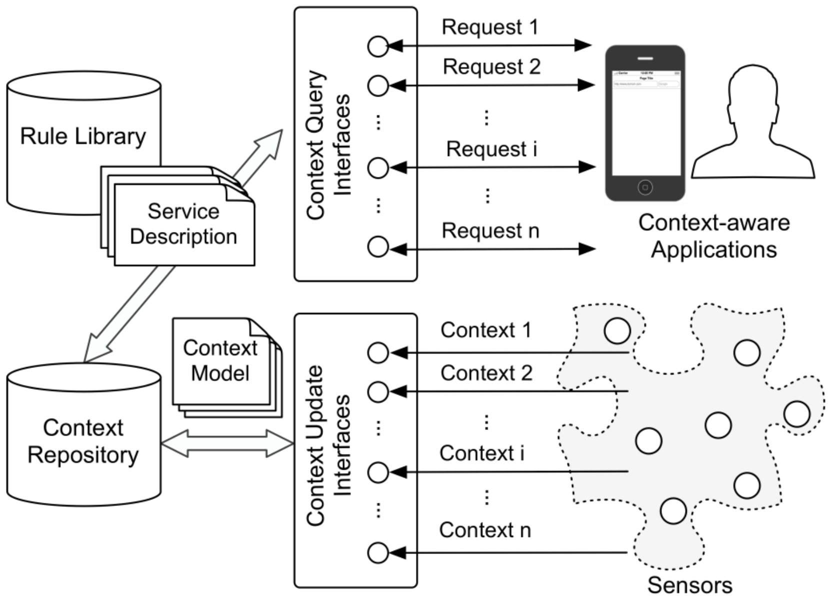


Figure 6-3 Context Integration for Context-aware Service Provisioning

In context-aware service end, the context integration is driven by events such as user request and context variation, as is shown in Figure 6-3. The diverse requests are modelled according to their functions, and the relevant application interfaces are provided accordingly. The context-aware adaptation architecture defines the functional components and the corresponding interfaces for each context request, either for context update or for context dissemination.

An example application interface for context data update is as below:

Method /context-server/update/context-path?context-items

An example application interface for context data query is as below:

Method /context-server/query/context-path?context-items

The service architecture defines the principles for encoding, addressing, and accessing of the context data as resources using Internet standards such as Uniform Resource Identifiers (URI), HTTP, XML and JSON between the mobile clients and service applications. In order to provide access to context information, the HTTP methods (i.e., GET, PUT, POST, DELETE) are used to encode the operations typically offered by context providers: retrieve using GET, notify with PUT, subscribe using POST, and unsubscribe with DELETE. The explicit use of the widely accepted standard and protocols can also enhance the interoperability of the system which interacts with the heterogeneous mobile and embedded devices.

By employing the modularisation design principle, the software components are separated clearly and the context data is integrated in a transparent way. The context update and distribution are then achieved with the diverse widely accepted protocols and standards.

6.2 Context Reasoning – From Low-level Context to High-level Context

As mentioned above, there is a semantic gap between the sensor data and useful context that is applicable to the context-aware computing systems. A lot of useful data may not be able to be observed with the sensors or other context sources directly (Beamon and Kumar, 2010). Some of the context items can only be inferred with the low-level sensor data for further use. Context reasoning is the procedure to deduce new contexts based on information about various other contexts (Saeedi et al., 2010).

The traditional method for context reasoning is based on IF-THEN logic, which uses fixed rules to infer higher level contexts. For this kind of method, the low-level context that is relevant to the higher level context should be identified in advance. Besides the rule-based method, some mathematical models can be used for context reasoning, such as DT, BN (Sekkas et al., 2007), kNN, Case-based Reasoning (Hong et al., 2009) and FL (Guan et al., 2006). The context reasoning module requires formal and structured context data that can be used in inference computation. In what follows, the context reasoning model and employed context reasoning method are discussed.

6.2.1 Context Reasoning in Context-aware System

The low-level context is sometimes too implicit to be used by the applications, and the high-level context may be more explicit and effective for the applications to understand. For example, if there is a task: when the user is sleeping in bed, turn off all electronic home facilities. It may be reasoned by low-level context as: (1) user is in bed, (2) bedroom is quiet, (3) bedroom light is turned off, and (4) time is late and it is user’s sleeping time. With all the conditions satisfied, there is a probability that the user is sleeping in bed, and the electronic home facilities can be turned off. The context reasoning module in a context-aware system can be described with Figure 6-4.

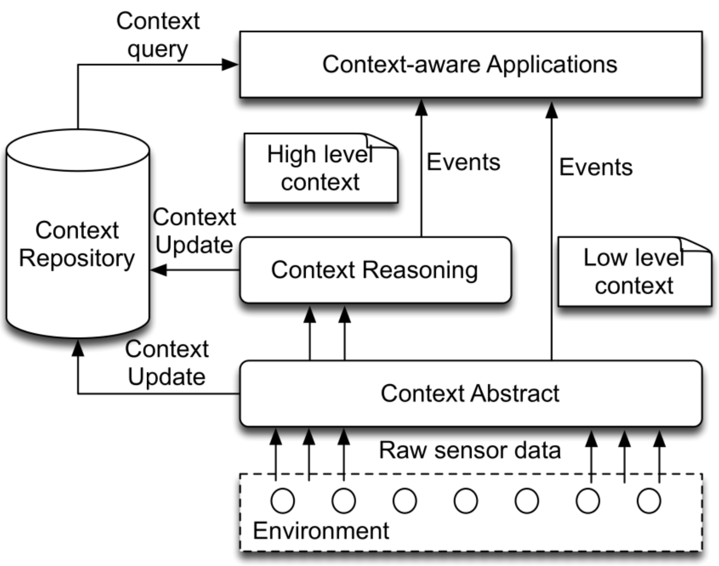


Figure 6-4 Context Reasoning in Context-aware Computing System

As shown in Figure 6-4, some low-level context can be used directly by the applications, and some others may be used to infer high-level context. Both low-level and high-level context relating to some context-aware applications are updated and stored in the context repository.

6.2.2 Rule-based Context Reasoning Methods

For the context reasoning, the relevant context items that are used as inputs in the reasoning should be identified or normalised first. For the rule-based method, the context reasoning is based on the rules defining the relationship between the high-level and low-level contexts. For some complicated mathematical methods, the relationship between the high-level context and low-level context is calculated by classification.

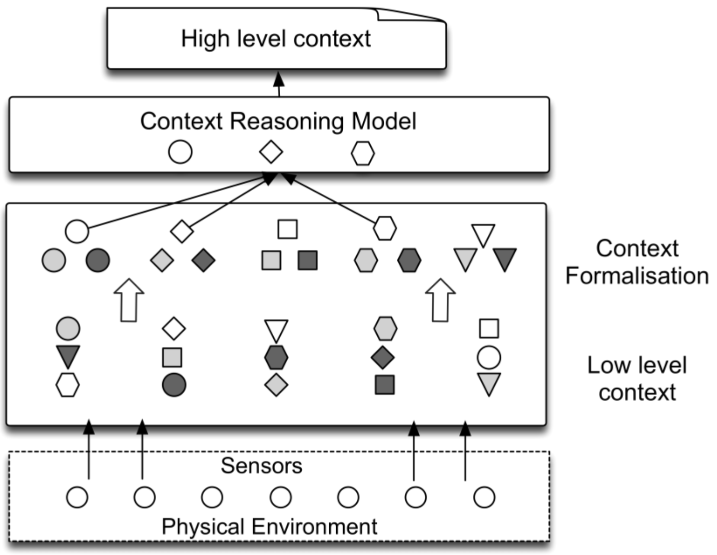


Figure 6-5 Fundamental of Context Reasoning

Just as is shown in Figure 6-5, the low-level context obtained with the sensors can be classified and normalised in a structured format. Then, the high-level context can be obtained with the context reasoning methods. In this section, a rule-based context reasoning method is introduced for simplicity and efficiency, which is given as follows.

Let the finite set $A=[A_1,A_2,...,A_n]$ be the context attribute with n $(n>1)$ items, and $a_i=[a_{1i},a_{2i},...a_{ni}]$ $(i>0)$ is an observation of A. Suppose finite set $H=[h_1,h_2,...,h_k]$ is the high-level context with $k(1>k)$ context values, and the value of H is determined by the low-level context attributes $A_1, A_2, ..., A_n$. The value of high-level context H can be inferred using A_1, $A_2, ..., A_n$ with a corresponding relationship $[av_{1i},av_{2i},...,av_{ni}]\rightarrow h_i$ $(1<i<k)$, where av_{1i} is a specific value of a_{1i}. The high-level context h_i can be deduced with the observation of related context value set a_i with equation:

$$(a_{1i} \text{ is } av_{1i}) \wedge ... \wedge (a_{ni} \text{ is } av_{ni}) \rightarrow (H \text{ is } h_i) \qquad (6\text{-}2)$$

For the rule-based context reasoning, if all observations a_i are assumed to be reliable, the high-level context obtained by the inference is considered to be reliable. However, sensors are sometimes inaccurate and it is necessary to incorporate accuracy estimation in the context reasoning. If $conf_i$ $(0\leq confi\leq 1)$ is defined with the confidence of the context source A_i, then (6-2) can be changed to:

$$((a_{1i} \text{ is } av_{1i}) \wedge conf_1) \wedge ...((a_{ni} \text{ is } av_{ni}) \wedge conf_n) \rightarrow (H \text{ is } h_i) \qquad (6\text{-}3)$$

The confidence of the value h_i can be evaluated with the set $(conf_1,conf_2,..., conf_n)$. According to certainty factor theory, the analogous confidence of add-aggregation antecedents can be evaluated with $min(conf_i), i=1,2,...,n$. Figure 6-6 illustrates the structure of the context reasoning with observed values.

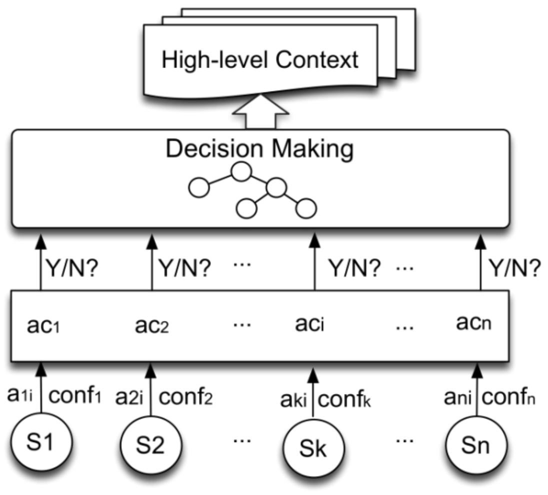


Figure 6-6 Diagram of the Context Reasoning

The rule-based reasoning is usually lightweight and easy to implement. However, the disadvantage of rule-based context reasoning is that it requires explicit definition of rules in its development. Thus, they are not flexible enough to deal with changing contextual situations. In addition to the rule-based method, the probability-based method is also widely used for context reasoning. Its strength is that it can change to adapt to situation change, which allows the system to be able to treat dynamically changing situations flexibly. However, a statistical model based method may need suitable quantity of context data and a training process is needed at the beginning of the system. It may also require more computation resources compared to light-weight and explicitly defined rule-based method.

As the focus of this research is context-aware service customisation, the rule-based context reasoning method is employed due to its simplicity, light-weightness, and explicit definition.

6.3 Context-aware Computing Service Provisioning

The ultimate purpose of context-aware systems is to provide a computing service that is able to reduce human efforts in applications with awareness of the contextual situation. This section provides methods for context-aware service customisation and discusses the key issues in context-aware application development. Treating context-aware service application as a decision information system, a Rule-Based Service Customisation (RBSC) strategy is proposed. The semantic distance-based rule matching method is employed for context-aware service decision making. Then, in order to make the service able to adapt to the dynamic situation, a rough set theory-based attribute reduction and rule generation method is employed for rule generation. Afterwards, critical issues in context-aware system development such as load balancing and context access control strategy are discussed.

6.3.1 RBSC Strategy

(1) The functional model for context-aware service customisation

In the context-aware service end, the proactive service is achieved by the context-aware service customisation engine according to the context variation. The service adaptation engine can be regarded as a decision information system which employs the context as input, and produces the appropriate information for service composition as output. For the proposed RBSC strategy, the rules supervising the service customisation are stored in the rule library, and rule generation and adjustment algorithms can create new rules and adjust the existing ones. The principle of the context-aware service customisation can be described with a functional model as shown in Figure 6-7.

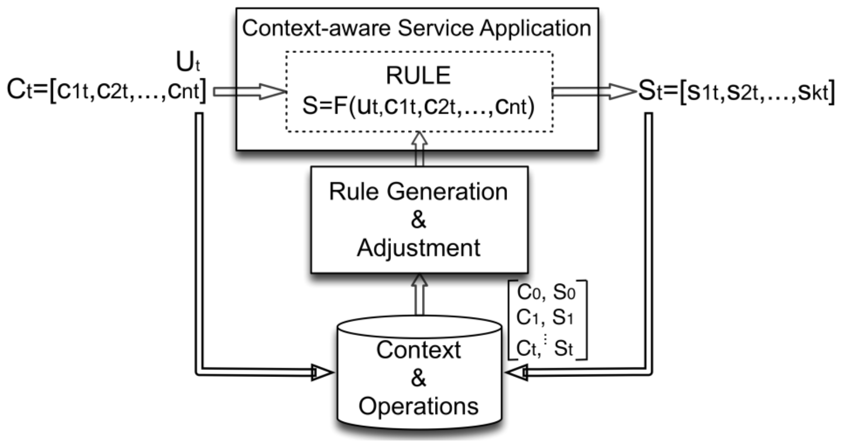


Figure 6-7 Functional Model for Context-aware Service Customisation

In context-aware computing systems, the computing service S_t at time t is determined by the rule function F:

$$S_t = F(U_t, C_t) = F(U_t, [c1_t, c2_t, ..., cn_t]) \tag{6-4}$$

In (6-4), U_t and C_t are the user requests and context at t and c_{1t}, c_{2t}, ..., c_{nt} are the context items in the context vector affecting the context-aware service. The history data of context information $C_H=[C_0, C_1, ..., C_n]$, user requests $U_H=[U_1, U_2, ..., U_n]$, and context-aware service $S_H=[S_0, S_1, ..., S_n]$ are stored in the context repository for further use. Of course, there are special cases where the context-aware application predicts the user's request and executes automatically. In this occasion, the *St* depends on only the context information *Ct*, which is common in some ambient intelligence applications.

(2) Rule-based method for service customisation

For context-aware systems, the service customisation method that generates computing service according to the relevant context items plays an important role in context-aware service provisioning. The commonly used approach that can be employed for the service customisation is the rule-based method. Just as is shown in Figure 4-3, the request is served with a service customisation engine that utilising the rules describing the logic between the service and relevant context items that are stored in the rule library. Thus, when an event happens, either user initiated requests or context value changes, the rule-based context-aware service customisation can be presented with expressions (6-5) and (6-6):

$$C : (U_t \vee c_{ti}) \rightarrow CR_t \tag{6-5}$$

$$S_C = F(U_t, CR_t) \tag{6-6}$$

where U_t, c_{ti} and CR_t stand for the user request, context change, and their related computing service. And F denotes the context-aware customisation rule that specifies the relationship between the service and related context. The application accesses the rule library to obtain the related context items, and provides the appropriate service with the customisation rule and values of the related context items.

Using the restaurant recommendation system as another example, the relevant context items can be categorised into time, location, traveling mode, weather, flavour, and rating. The table for related context and customisation rules of the context-aware recommendation are defined as shown in Table 6-1 and Table 6-2.

Table 6-1 Table of Context Related to the Restaurant Recommendation

Services	Context Items						
	Item 1	Item 2	Item 3	Item 4	Item 5	Item 6	…
Service 1	…	…	…	…	…	…	…
Service 2	…	…	…	…	…	…	…
…	…	…	…	…	…	…	…
Restaurant recommendation	Time	Location	Traveling mode	Weather	Flavour	Rating	…
…	…	…	…	…	…	…	…

Table 6-2 Customisation Rules for the Restaurant Recommendation

Rules	Related context						Services (Restaurants)
	Time	Location	Traveling mode	Weather	Flavour	Rating	
Rule 1	Midday	In 20 minutes	Walking	Sunny	British food	Above 4 stars	Ra, Rb, Rc
Rule 2	Midday	In 10 minutes	Walking	Raining	French food	Above 3 stars	Rd, Re, Rf
…	…	…	…	…	…	…	…
Rule i	Afternoon	In 20 minutes	Driving	/	Chinese food	Above 4 stars	Rg, Rh, Ri
…	…	…	…	…	…	…	…

Using the rules defined in Table 6-2, the users can be provided with a list of nearby restaurants that fit users' flavours and are of high rating according to the service customisation rules that consider users' traveling mode and the weather conditions.

(3) Semantic distance-based rule matching – principle and algorithm

In the context-aware service engine, the proposed strategy employs the semantic distance-based rule matching method for computing service adaptation, which compares the current context and the adaptation rules to provide the appropriate computing service. Let *C* be the attribute vector of context items, *U* be a sample of context value, and *V* be the rule vector. The distance between vectors can be calculated with the Manhattan Distance, which is also named Taxicab Geometry (Krause, 1987). In this research, Manhattan Distance is normalized to compute the distance of context vectors *dist(U,V)*, which is defined as follows:

$$dist(U,V) = \sum_{i=1}^{n} w_i \frac{|u_i - v_i|}{Range(c_i)} \tag{6-7}$$

Formula (6-7) defines the approximation of context value vectors *U* and *V*, where w_i ($0<w_i<1$, $\Sigma w_i=1$) and *Range*(c_i) are the weight and value range of the i^{th} attribute in *C* - c_i. The Manhattan Distance-based rule matching method may be suitable for the numeric context values. For non-numeric data, the difference between the values can be described with semantic distance, and researchers have already proposed many different methods of measuring the semantic distance (Benabderrahmane et al., 2010). In this investigation, the Generalised Cosine-Similarity Measure (GCSM) by Ganesan et al. (2003) is employed to compute the semantic distance between context sets, which is defined as follows:

Definition 1 Lowest Common Ancestor (LCA) LCA denotes the common ancestor of the maximum depth of two concepts in the ontology.

Definition 2 Generalised Cosine-Similarity Measure (GCSM) If c_1 and c_2 are two concepts in a tree-hierarchy of indexing terms and *depth*(c_1) and *depth*(c_2) are their depth in the hierarchy, the GCSM similarity between them is:

$$GCSM(c_1,c_2) = \frac{2 \times depth(LCA(c_1,c_2))}{depth(c_1) + depth(c_2)} \tag{6-8}$$

For non-numeric data, (6-8) can be used to calculate the distance of context values which is between 0 and 1. Meanwhile, the value of $GCSM(c_1,c_2)$ $(0<GCSM(c_1,c_2)<1)$ can present the approximation degree of concepts c_1 and c_2. Then,

$$dist(c_1,c_2) = 1 - GCSM(c_1,c_2) \tag{6-9}$$

can be used to present the semantic distance between concepts c_1 and c_2. Therefore, the formula calculating the distance of context vector and rule vector is:

$$dist(V,R)=\sum_{i=1}^{n} w_i.dist(v_i,r_i)=\sum_{i=1}^{n} w_i.(1-\frac{2\times depth(LCA(v_i,r_i))}{depth(v_i)+depth(r_i)}) \qquad (6\text{-}10)$$

where w_i is the weight of context item c_i, v_i and r_i are the value of context vector V and rule vector R on attribute c_i, and $dist(V,R)$ is the distance between vector V and vector R. Workflow of the semantic distance-based rule matching algorithm is given as follows.

Algorithm 1 - Semantic Distance-based Rule Matching Algorithm

---

Inputs of the algorithm:

Context – the current context values
Rule – rule set of context-aware adaptation
Reduction – result of attribute reduction, namely context related to the computing service
Weight – the weight of context attributes

Outputs:

Matched_rule – the matching rule

Steps of algorithm 1:

```
Step1. Initiate minimum distance min_dist =1 and rule ID min_ruleid;
Step2. Calculate each Rule[i] in rule library;
Step3. Calculate each attribute item in reduction Reduction[j];
    if(Reduction[j] is numeric data)
        dist+=Weight[j]*abs(Context[j]-
        Rule[i].getValue(Reduction[j].id))/Reduction[j].getRange();
    else if(Reduction[j] is non-numeric data)
        dist+=Weight[j]*(1-GCSM(Context[j],Rule[i].getValue(Reduction[j].id)));
Step4. if(Items in Reduction is finished){
            if(dist<min_dist) min_dist =dist, min_ruleid=i, dist=0;
             Go to Step2;
      } else
           Go to Step3;
Step5. The minimum distance is min_dist, and the matching rule
        Matched_rule=Rule[min_ruleid].
```

---

(4) Temporal complexity of the rule matching algorithm

The temporal complexity of the algorithm is decided by number of rules, number of attributes in the rules, and data type and structure of attributes. The performance is evaluated by simulation on a Lenovo Laptop with Intel Dual-Core T6600 2.2GHz, FSB Speed 800M, L2 Cache 2M, and 4GB DDR3 RAM.

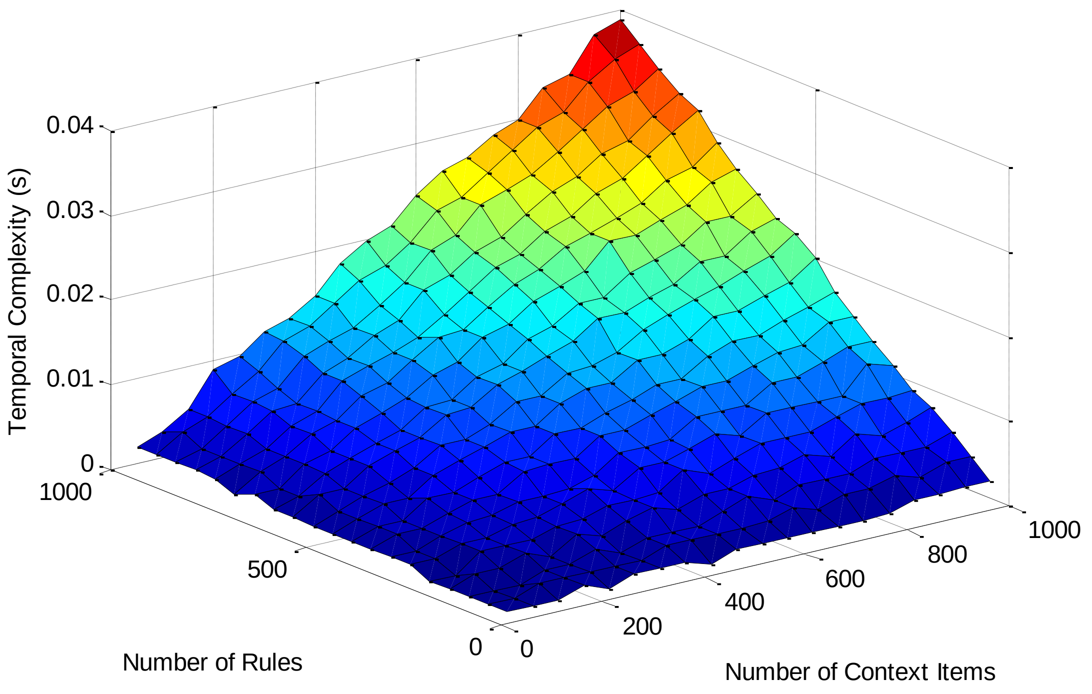


Figure 6-8 Temporal Complexity of Rule Matching Algorithm

In the simulation, there are equal numbers of numeric data and ontology terms in all the rules. Figure 6-8 gives the time consumption of the algorithm when the number of rules increases from 50 to 1000 and number of context items increases from 50 to 1000. Figure 6-9 shows the results for rules with 100, 300, and 500 attributes and the number of rules increase from 50 to 1000.

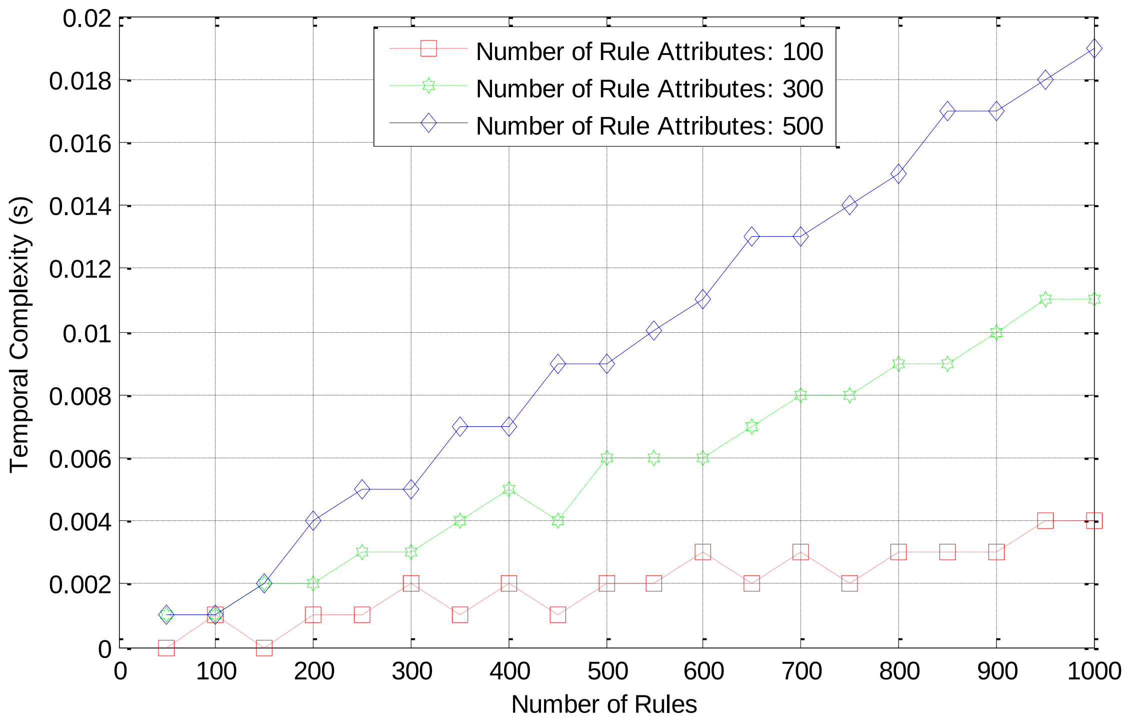


Figure 6-9 Speed of the Rule Matching with Specific Quantity of Attributes

From the simulation results, time consumption of the algorithm increases nearly in line with the number of rules and number of attributes in the rules. The algorithm is lightweight which is appropriate for the resource limited mobile devices. Since the number of rules may not be large for practical applications, the real-time performance may not be a problem. The performance of rule matching algorithm is evaluated by practical application in Chapter 8.

With the pre-defined rules specifying the service and the relevant context items, the system can provide the appropriate computing service with the rule matching method automatically. However, as the pre-defined rules may not be suitable to everyone using the context-aware systems for different use scenarios, the system needs to be able to generate new adaptation rules and adjust the existing rules automatically. Thus, in the context-aware adaptation systems, attributes reduction and rule generation methods are required to allow the capability of self-learning for rule generation and adjustment to deal with sophisticated context situations.

6.3.2 Rule Generation Method in RBSC Strategy

(1) Fundamentals of context-aware rule generation

In the rule-based context-aware systems, the default rules are normally defined by the developers based on their knowledge of the application scenario. However, for some cases of context-aware applications, the influencing context items and degree of their effects are usually multi-faceted, which are difficult to be determined directly. Therefore, the performance of context-aware service customisation may be constrained by the default rules defined based on the developers' knowledge. This may result from the following reasons:

- The context-aware computing service may be influenced by some context items that are not included in the related context *Cr*, and new rules need to be generated.
- Some context items in the related context *Cr* for a particular context-aware computing service may be redundant, and existing rules need to be removed.
- The degree of the context affecting the computing service may change, and the rules may need to be revised to better adapt the service to users' preference.

As discussed above, it is significant to allow the system to be able to determine the rules that are appropriate for context-aware service customisation. However, the context information is characterised by large quantity, interrelated and heterogeneous data sources and data formats. In addition, some uncertain context data may be generated in the context acquisition and interaction between mobile device and server. So, the method to determine the appropriate context items for the supervision of the context-aware service customisation towards proactive computing service is important for context-aware systems.

As stated in Chapter 4, the history context and corresponding service is stored in the context repository, which can be explored to derive new rules for computing service customisation.

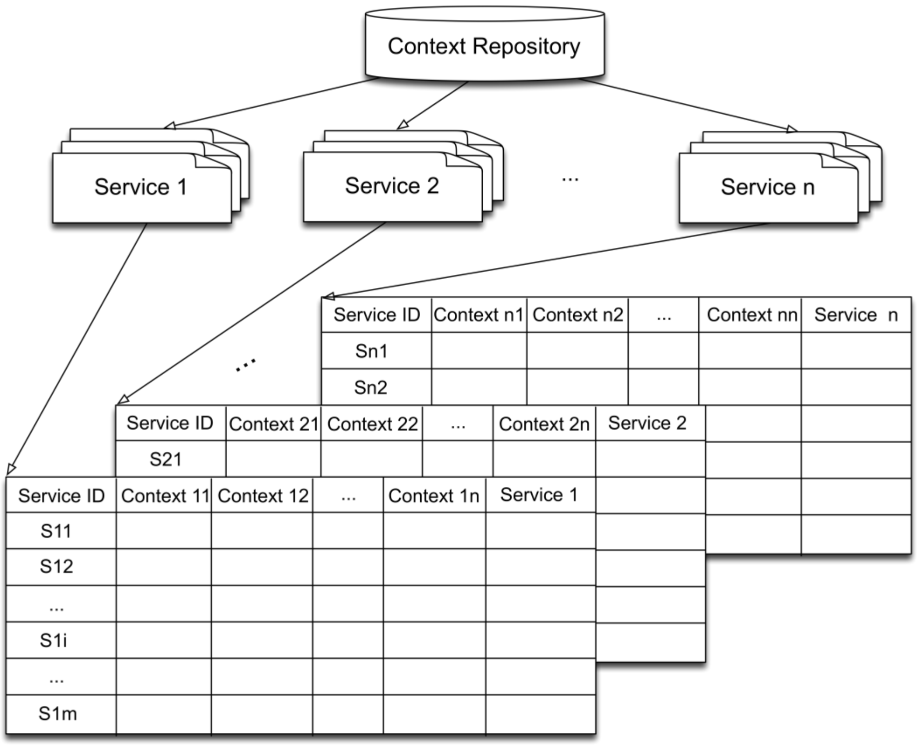


Figure 6-10 History Context and Corresponding Service in Context Repository

In order to derive new rules with the history context, the context data stored in the database can be organised and accessed with a data structure as is shown in Figure 6-10. The context-aware service can be classified into S_1, S_2, ... , S_n. For each context-aware service S_i, the related context items *Context* i_1 - *Context in* and the corresponding service *Service i* are recorded in the context repository. With all the S_i (S_{i1}, S_{i2}, ..., S_{in}) records in the context repository, the history data of the service and the corresponding context data can constitute the following matrix:

$$\begin{bmatrix} C_{i1\_1} & C_{i2\_1} & \cdots & C_{ii\_1} & \cdots & C_{in\_1} & S_{i1} \\ C_{i1\_2} & C_{i2\_2} & \cdots & C_{ii\_2} & \cdots & C_{in\_2} & S_{i2} \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ C_{i1\_i} & C_{i2\_i} & \cdots & C_{ii\_i} & \cdots & C_{in\_i} & S_{ii} \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ C_{i1\_m} & C_{i2\_m} & \cdots & C_{ii\_m} & \cdots & C_{in\_m} & S_{im} \end{bmatrix}$$

With this matrix, the rules for context-aware service customisation can be obtained with some mathematical models by determining the significant context items.

Essentially, rule generation in the context-aware system is a knowledge discovery system, and the rule is the knowledge to supervise the context-aware service provisioning. Some mathematical models can be considered, such as DT, BN, HMM, FL, SVM, and kNN, and Rough Set Theory (RST) (Pawlak, 1982; Pawlak and Skowron, 2007; Huang et al., 2011). As the RST is a mathematical theory for dealing with uncertain data possibly caused by imprecise measurement, network latency, and sampling errors (Suresh et al., 2012), it is appropriate to be employed for context-aware adaptation rule generation. This work uses the same concept as the Rough-Fuzzy method discussed in section 2.4.3.2, which utilises the rough set to determine user preferred keywords in Web information retrieval. The rule generation method in this work uses the RST to determine the attributes making greater contribution to the decision making of context-aware service and generate rules accordingly.

(2) Rule generation method based on Rough Set Theory

For the rule generation in this investigation, suppose there are n different contexts that constitute condition attribute set $A=\{C_1,C_2,\dots,C_n\}$. $CV=\{V_1,V_2,\dots,V_n\}$ can be used to denote an observation value of A, where V_i is the value of C_i in the sample. Then, set $U=[CV_1,CV_2,\dots,CV_m]^T$ constituted with the context vectors can be regarded as the discourse domain. Therefore, the decision information system is thus built up if the services requested are the decision attributes.

Definition 3 Information System $I=(U, A, V, f)$ is an information system provided U is a non-empty finite objects set, A is non-empty finite attributes set, V_a is the value range of attribute a and $V = \cup_{a\in A} V_a$ is the union of attribute domains, and $f: U\times A\rightarrow V$ is an information function so that for any $x\in U$ and $a\in A$, $f(x,a) \in V_a$.

Definition 4 Decision Information System (DIS) Information system $I=(U,A,V,f)$ is a decision information system if I satisfies conditions: $A=C\cup D$ and $C\cap D=\emptyset$, where, C is the condition attribute and D is the decision attribute.

Definition 5 Indiscernibility Relation Given $\forall P \subseteq A$, the equivalence relation $IND(P)$ is defined the indiscernibility relation, where

$$IND(P) = \left\{(x,y)\in U^2 \,\middle|\, \forall a \in P, a(x) = a(y)\right\} \tag{6-11}$$

As context may contain instantiated data rather than numeric only, $IND(P)$ can be defined as:

$$IND(P)=\{(cv_i,cv_j)\in CV^2 \mid \forall a\in P, a(cv_i)=a(cv_j)\vee class(a(cv_i))=class(a(cv_j))\} \quad (6\text{-}12)$$

where $P \subseteq A$, $a(x)$ is the value of attribute a in vector x, and $class(v)$ denotes the class that v belongs to.

Definition 6 Reduction and Core Given $P\subseteq A$ in an information system I, the $IND(P)$ divides object set U into k equivalence classes, denoted $U/P=\{X_1,X_2,...,X_k\}$. Suppose $Q\subseteq P$, Q is independent and $IND(Q)=IND(P)$, then Q is a Reduction of P. If $RED(P)$ is all the reduction of P, $CORE(P)=\cap RED(P)$ is the Core of P.

Definition 7 Discernibility Matrix Given a DIS, $C=\{a_i|i=1,2,\dots,m\}$ and $D=\{d\}$ are the condition attributes and decision attributes, $U=\{x_1,x_2,\dots,x_n\}$ is the discourse domain, $a_i(x_j)$ is the value of sample x_j on attribute a_i. The discernibility matrix $DM_{i\times j}$ can be defined as:

$$DM(i,j)=\begin{cases}\{a_k \mid a_k(x_i)\neq a_k(x_j)\vee class(a_k(x_i))\neq class(a_k(x_j))\}, d(x_i)\neq d(x_j)\\ 0, d(x_i)=d(x_j)\end{cases} \quad (6\text{-}13)$$

where $i,j=1,2,\dots n$.

In a decision information system, the dependence of decision attributes D on condition attributes is different, and some condition attributes may be redundant. Admittedly, it makes no difference if the redundant attributes are eliminated. The task of rule generation is to derive the significant attributes $RED(C)$ to supervise the decision making of the context-aware computing service. There are several methods can be used for the attribute reduction based on rough set theory. In this investigation, the discernibility matrix-based reduction is employed because of its low computational complexity. Workflow of the rule generation algorithm is given as follows.

Algorithm 2 - Attribute Reduction and Rule Generation Algorithm

---

Inputs of the algorithm:

$CV = (CV_1,CV_2,\dots,CV_n)$ - The context observations;

$A = (C_1,C_2,\dots C_n)$ - Context attribute set;

O - Record of user operations or service.

Outputs:

Rules - context-aware rule set;

Core - the core;

Reduction - result of attribute reduction.

Steps of algorithm 2:

Step1. Calculate discernible matrix DM of decision table consists of CV and O, initiate reduction set Reduction=Φ;

Step2. Add the Core (The attributes in the DM element whose cardinal number of is 1) to Reduction, and eliminate the attribute items that contain the Core attribute;

Step3. Calculate the attribute frequency of the rest attribute items using the function $g(a_i)=\sum_{j=2}^{n}(num_{a_i}/j)$ *(where* num_{a_i} *is the number of* a_i *in DM, j is the number of attributes in the attribute item that contains* a_i*, and n is the maximum value of j);*

Step4. Add the attribute of highest frequency a_i *to Reduction, eliminate the attribute sets that contain* a_i*;*

*Step5. If (DM != **0**) Go to Step3; else go to Step6;*

Step6. Create the Rules according to the Reduction.

With this algorithm, the context items that are significant to particular context-aware computing services can be determined. Therefore, the user satisfaction of the computing service can be raised and complexity and cost of decision making can be reduced.

(3) Temporal complexity of the rule generation algorithm

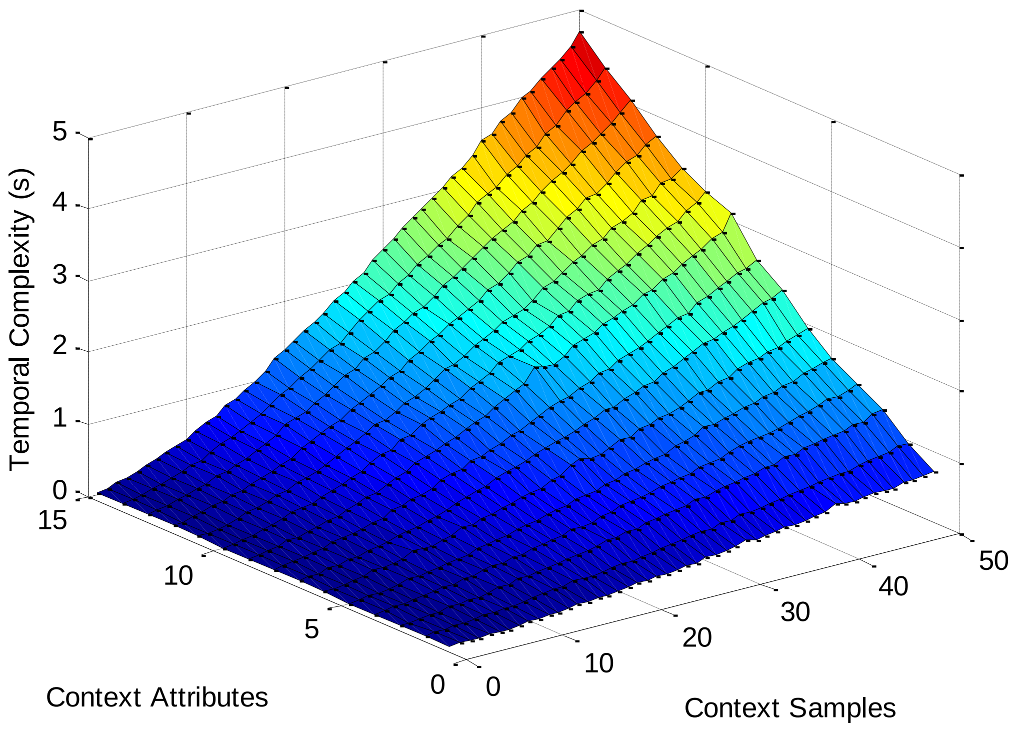


Figure 6-11 Temporal Complexity of Rule Generation Algorithm

The temporal complexity is evaluated by simulation with the Lenovo Laptop as well. As the relevant context items may not exceed 15 for a particular context-aware computing task, the quantity of context items is set from 1 to 15 in the simulation. Also from the practical application point of view, the quantity of context records is set from 1 to 50. Then, the result is as shown in Figure 6-11 and Figure 6-12.

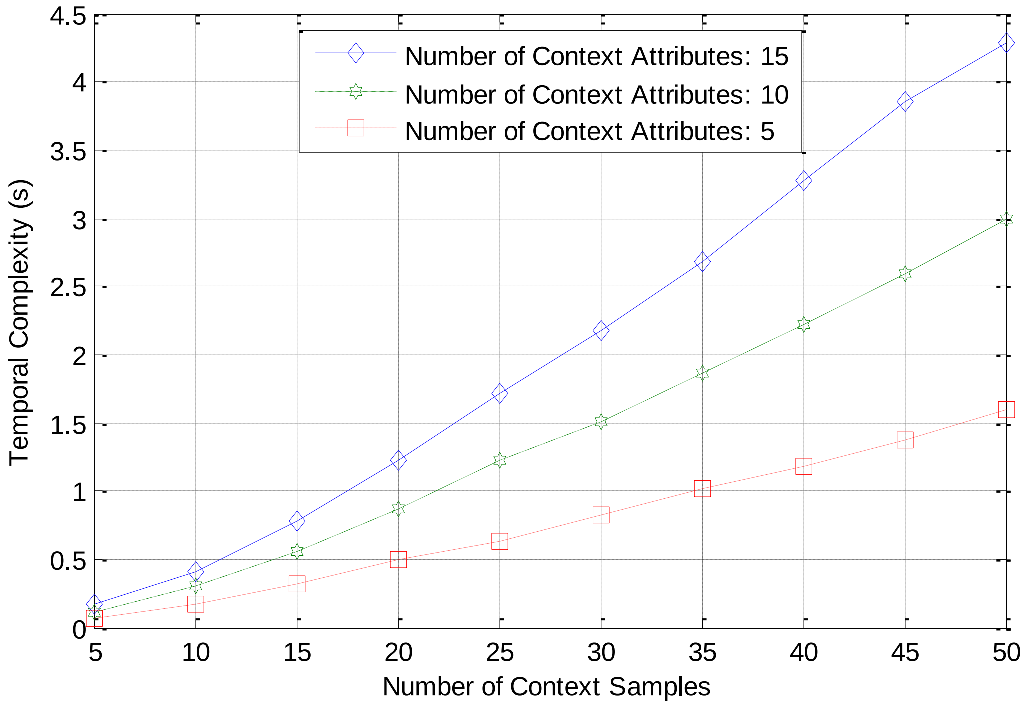


Figure 6-12 Speed of the Rule Generation with Specific Quantity of Attributes

Figure 6-11 gives the trend of how temporal complexity of the algorithm varies with the number of context samples and condition attributes. The number of context samples increases from 1 to 50, and the number of context attributes increase from 1 to 15. From the figure, it is easy to find that the time consumption of the algorithm increases with the quantity of context samples and the quantity of context condition attributes. Then, Figure 6-12 gives an even clearer view of the speed of the algorithm with the number of context attributes 5, 10, 15, and context samples from 1 to 50. According to the curve, the speed of the algorithm changes fast with the increase of context samples and context attributes. Therefore, to select the significant context attributes and the appropriate number of context samples is very important for rule generation.

From the discussion in Chapter 2, most of the investigations into context-aware computing systems in the early years were based on IF-THEN-based systems. Some new proposed methods may focus on method design and theoretical study. It lacks experimental study which puts the proposed methods into practical use and detailed evaluation. In this investigation, the above figures just give the trend; the real performance of the context-aware service end may be dependent on the computational power of the server. Practical experimental study is needed to demonstrate the feasibility of the proposed method in both effectiveness and efficiency. In order to prove the effectiveness of the rule matching algorithm and rule generation algorithm described above, the experimental studies are carried out in Chapter 8.

6.3.3 Load Balancing of Distributed Context-aware System

In context-aware mobile systems, the user requests are carried out with the distributed mobile devices, while the context-aware customisation is centralised computation accomplished in the service end. Since there is significant difference in computation power between the heterogeneous mobile devices, the computation load of the components in the system should be optimised to enhance performance of the whole system. The load balancing for a context-aware mobile system is required in the design and development for the following reasons:

- Computation power of the distributed computing terminals differ greatly
- Complexity of context acquisition for various contexts vary considerably
- Complexity of service customisation tasks are different
- Throughput of wireless network may restrict large amount data exchange
- Concurrent requests from numerous users may congest the service end

Therefore, it is necessary to distribute the computation power optimally in order to efficiently utilise the computation resource and guarantee the performance of the system. In this investigation, the distribution of computation intensive tasks in the system follows the strategies below:

- Complex computation for context feature extraction is finished by mobile client or embedded devices, such as barcode scanning and human behaviour recognition
- Context abstraction for periodic or high-speed signal can be finished at mobile or embedded client end
- Context data from Internet data provider is parsed at the mobile client end
- Context-aware service customisation is finished at the service end
- Rule generation for supervision of service customisation is finished at the service end

With the above strategies, the computation tasks are distributed to the mobile devices, embedded clients, and the servers as shown in Figure 6-13. Two objectives can be achieved by the above design strategies:

(1) To reduce large amounts of data exchange between mobile clients and server, and

(2) To reduce the concurrency of complex computations at the service end.

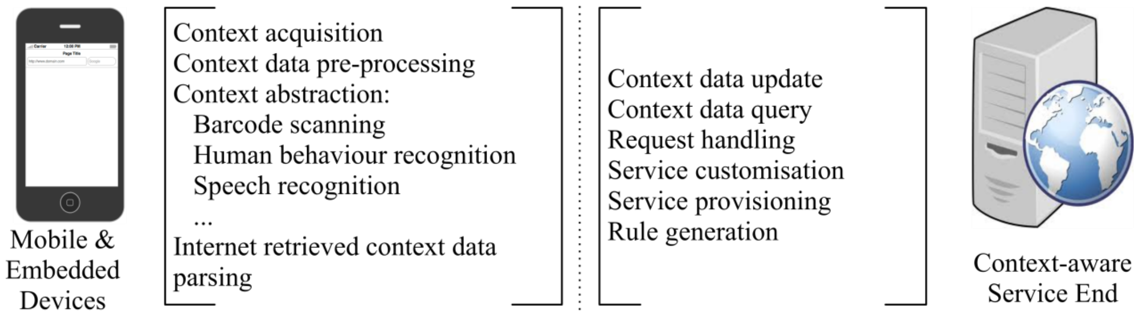


Figure 6-13 Load Balancing in Context-aware Mobile Systems

Therefore, the risks of network congestion and heavy computation load on centralised server are relieved to some extent, and performance of the distributed system can be improved.

6.3.4 Context Access Control in Context-aware Systems

In mobile based context-aware systems, the context data may contain a user's private information which is important to the context-aware applications. Admittedly, the private information needs to be protected from unauthorised access. Thus, the method to use the context data flexibly whilst guaranteeing the safety of sensitive data is critical to the system.

To go on with this topic, it is important to gain a clear view of what makes the context data in a pervasive environment any different from other computing systems. Langheinrich (2001) has identified four key motivators: ubiquity, invisibility, sensing, and memory amplification, which can be regarded as significant features of context-aware systems.

It has already been observed by researchers that security, privacy, and trust raise considerable challenges in the area of mobile phone sensing (Miluzzo et al., 2010a). Sheikh et al. (2008) proposed a QoC-based privacy policy to protect users' privacy by specifying the QoC indicators and schemes for quantitative expression. In project Aware Home (Covington et al., 2000), the access control mechanism employed Generalised Role-based Access Control (GRBAC) - an improved version of traditional RBAC presented by Sandhu et al. (1996). The GRBAC in the Aware Home project enhances traditional BRAC by incorporating the notation of object roles and environment roles. Smailagic et al. (2001) controlled the privacy of location information with a simple rule-based set theory, which specified the authorisation based on one of the four visibility modes: visible to all, invisible to some, visible to some, and invisible to all, and each rule builds a table of users who are allowed or not allowed to access the location information.

In this investigation, a grouped rule-based context access control (GRCAC) strategy is designed to protect the privacy of context data. The GRCAC strategy is characterised by two characteristics: (1) Group-based classification of entities; and (2) Hierarchical structure. In this context access control policy, when, how, and to whom context data will be disclosed are essential issues to be explicitly defined.

(1) Structure of GRCAC

In the first place, the entities that provide, consume, and process context in the proposed context access control are listed and described as follows:

- Context provider: Users or other entities provide context data for the system.
- Context consumer: Users or other entities request to access the context data.
- Applications: Applications that make use of the context data.

For the management of context data access, the system is divided into three levels: administrator level, individual level, and basic level.

- Administrator level: The administrator level classifies the context consumers and context items into groups and defines their access permission.
- Individual level: The individual level allows the individual users to define the permission of their data used as context.
- Basic level: The basic level defines the normal distribution of context data without special consideration of privacy protection.

(2) Group definition for context access control

The user members in the system are categorised into groups for context data access control. When the context consumer request some context data, the request will be evaluated with the privacy rules considering the group the context consumer is in. The classification of the group depends on the use scenario of the application. For example, in a smart home system, the context consumer C_C may be categorised into the groups of:

$$C_C \rightarrow \{M_f, M_n, M_c, M_t, M_u\} \qquad (6\text{-}14)$$

where the items M_f, M_n, M_c, M_t, and M_u in the expression denote family members, neighbours members, community members, city members, and unknown members. Meanwhile, the context providers C_P are also categorised into different groups. For example:

$$C_P \rightarrow \{U_d, U_e, U_b, U_p, U_s\} \qquad (6\text{-}15)$$

where the items U_d, U_e, U_b, U_p, and U_s in the expression denote user's devices, user's environment, user's behaviour, user's preference, user's social networks, etc.

The privacy of the context data can be defined by the either the administrator or the individual users for context access control. The categorization of the context providers and context consumers into groups has enriched the granularity of access control and facilitated the computation during evaluation of context requests. Requests from the context consumer group matching the rule will be permitted to access context from corresponding context provider groups, and those do not match will be denied.

(3) Grouped rule based context access control model

The group-based RBAC strategy categorises the context providers and context consumers into groups to facilitate the context data access control. The groups of context consumers hold different permissions to access some particular groups of context items, and the context providers are permitted to be accessed by particular groups of context consumers

As shown in Figure 6-14, the model of proposed GRCAC consists of three entities: context consumer and groups (C_C), context provider and groups (C_P), and permission (P). For simplicity, the context consumer denotes a user or an application that uses the context data. The context provider denotes a user or device which provides the context data for the system to use. Permission is the mapping table defining the rules of the C_C to C_P that is set by the administrator or context provider. To give an explicit explanation of the strategy, the methods for context access control in the two case studies in Chapter 8 are given as follows:

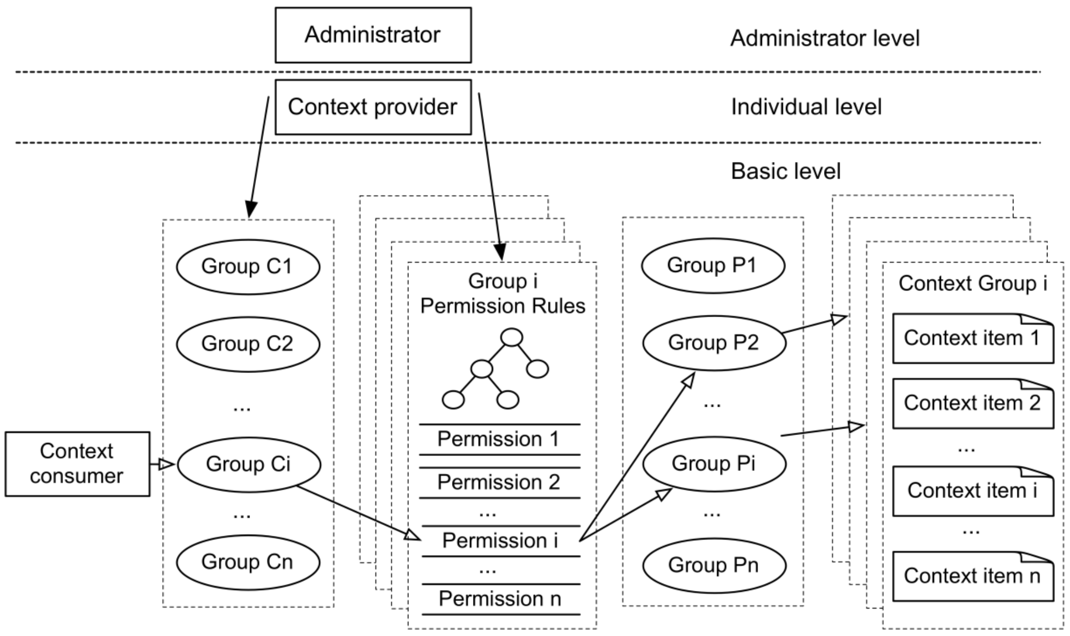


Figure 6-14 Context Access Control Model

Example 1: Smart home system

As discussed above, the context providers and context consumers are categorised into the groups as expressions (6-14) and (6-15), and the rule lists the permission of the mapping

$C_C[i] \rightarrow C_P[j]$. For instance, when $i=2$, $j=3$, the context consumer *Mn* requests context *Ub* of some users. If the user sets the permission to allow the *Mn* to access the *Ub* using some application, the neighbour members are allowed to access the user's behaviour. Otherwise, if by default the system does not allow the *Mn* to access the *Ub*, the neighbour members are denied access to the user's behaviour.

Example 2: Mobile Exam System user authentication

For the authentication of MES, the students login to the system with username, password, exam ID. The students in the class having the exam *Ei* are in a group C_{Class_i}, and classes not having the specific exam belong to others groups. As the exam is held in a particular classroom, users in the classroom (or building if position information is not precise enough) can be classified into group $U_{Classroom\_j}$. The users that are not in the classroom (or building) are not permitted to login to the exam *Ei*. On the other hand, the exam *Ex* for another class is another group of context information. The permission rules can be presented with the DT in Figure 6-15.

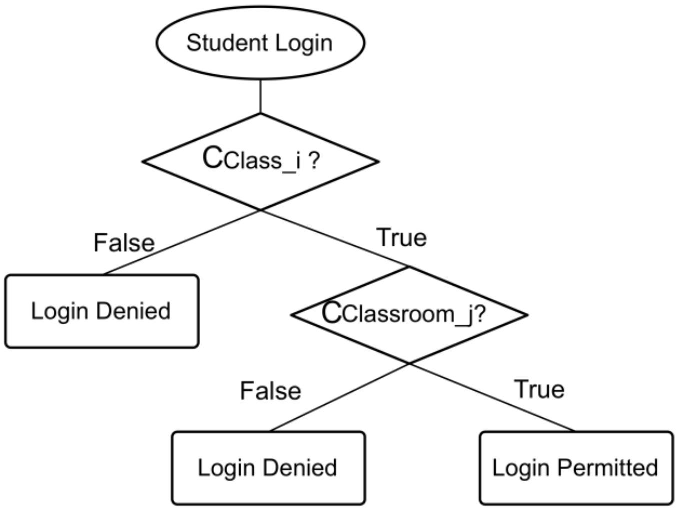


Figure 6-15 Permission Rules for MES Authentication

From Example 1 and Example 2, the context data can be classified and protected by the access control rules with the proposed method. However, it is necessary to emphasise that the GRCAC is just an access control model to protect the privacy of context data, not a complete security solution.

This section provides the critical methods for the context-aware service provisioning. Based on the context-aware service customisation model, the rule-based service customisation and rule generation methods are presented systematically. Then, the context access control strategy is designed to protect users' sensitive data being used as context data. In what follows, the key issues of context-aware mobile application evaluation are discussed.

6.4 Summary

This chapter focuses on context data computation and context-aware service provisioning. Firstly, methods for context integration and context reasoning are provided to bridge the gap between raw sensor data and high-level context applicable to the system. Then, the RBSC strategy is proposed, which employs the semantic distance-based rule matching method for context-aware service decision making, and a rough set theory-based method for rules generation using history context. In order to protect the privacy of users' context, a grouped rule-based access control strategy is proposed. The proposed methods are implemented and evaluated in case studies in Chapter 8.

Chapter 7 Evaluation of Context-aware Mobile Systems

Evaluation is a significant step for the design and implementation of computing systems, while the evaluation of the context-aware systems is endowed with even richer meanings because of the involvement of context data. In this chapter, the requirements and characteristics of the context-aware mobile applications are determined, and the important parameters and indexes evaluating the mobile applications are summarised in an evaluation framework, which can be used to supervise the development of context-aware mobile application as design guidelines.

7.1 Characteristics of Mobile Device-based Applications

Because of the resource limitation of mobile devices, the mobile applications are expected to be simple and lightweight. The resource limitation of mobile devices is mainly reflected in computation power, communication bandwidth, and HCI capability. Therefore, some novel methods and techniques are involved to improve the HCI capability of the mobile application or to efficiently explore the computation and communication resources. In summary, the mobile device-based applications are characterised by the following characteristics:

- Simple and easy to use
- Lightweight and low resource consumption
- Friendly and attractive in user interface
- Weak in data input and information presentation
- Rich in built-in sensor and hardware module resources
- Enhanced user interaction with various sensing techniques
- Open to Internet resources with access to wireless network
- Used handheld and sometimes on the move

These characteristics distinguish the context-aware mobile application from their traditional counterpart based on PCs. Only with the comprehensive consideration of these characteristics can the evaluation guidelines for context-aware mobile applications be determined.

7.2 Evaluation of Context-aware Mobile Applications

For evaluation of traditional computing systems, the usability of the application or system is regarded as an important indicator. It reflects the ease and efficiency of use, and is a significant part that influences the end user experience. The definition of usability by ISO 13407 standard (1999) on Human-Centred Design Process for Interactive Systems is: *extent* *to which a product can be used by specified users to achieve specified goals with effectiveness, efficiency and satisfaction in a specified context of use*. The context awareness itself is a technical and theoretical approach to improving the usability of computing systems.

For the evaluation of usability of computing systems, Preece et al. (2002) summarised the usability goals: effectiveness, efficiency, safety, utility, learnability, and memorability to guide the interaction design research. When it comes to the mobile device-based context-aware systems, the evaluation is endowed with much richer contents in terms of the acquisition and utilisation of context data. These special requirements are summarised below:

- Unobtrusiveness in context sensing
- Fast and accurate context acquisition
- Efficient and interoperable context distribution
- Effectiveness of context-aware service customisation
- Security and privacy of sensitive context
- Learnability based on context data

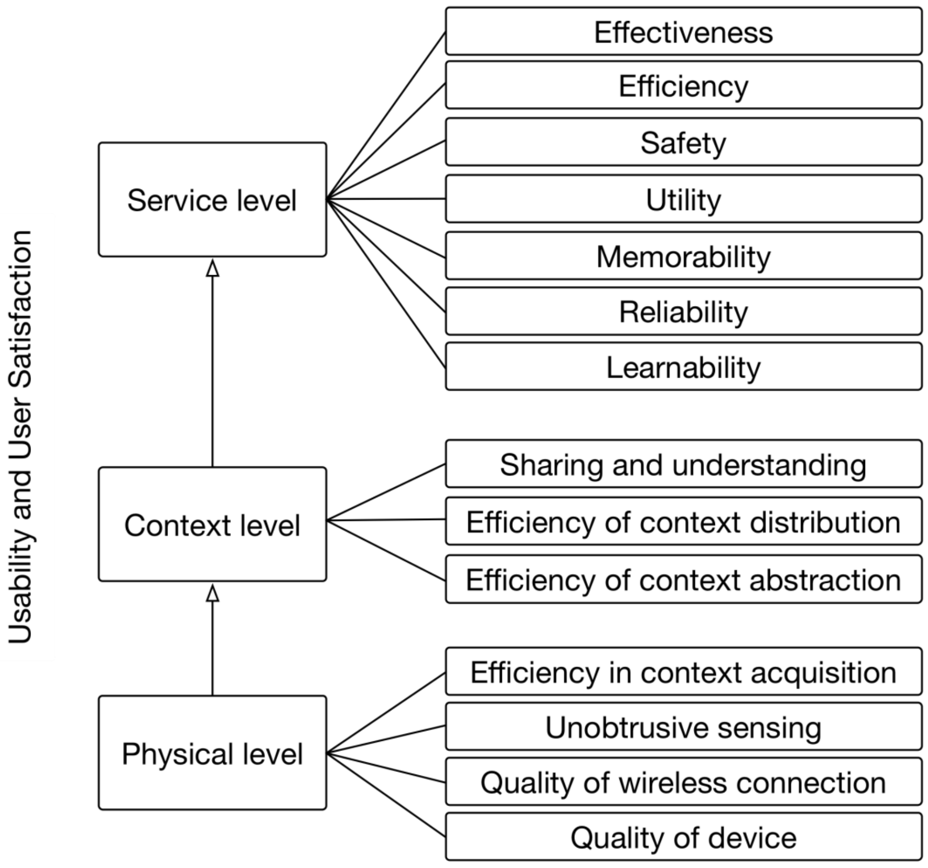

Usability and User Satisfaction
Service level
Context level
Physical level
Effectiveness
Efficiency
Safety
Utility
Memorability
Reliability
Learnability
Sharing and understanding
Efficiency of context distribution
Efficiency of context abstraction
Efficiency in context acquisition
Unobtrusive sensing
Quality of wireless connection
Quality of device

Figure 7-1 Evaluation Framework of Context-aware Mobile System

Due to the specificity of context-aware mobile applications, the evaluation of usability and user satisfaction may involve more abundant content compared to traditional systems. Accordingly, the evaluation framework can be described in a hierarchical structure with three levels corresponding to the layers in the context-aware system architecture: system level, context level, and physical level. The diagram in Figure 7-1 gives an evaluation framework of a context-aware mobile computing system.

- Service level

The service level may contain the parameters mainly at the service and application level describing the usability and user experience of the system. As mentioned above, the parameters are effectiveness, efficiency, safety, utility, memorability, reliability, and learnability. The performance described with these parameters can be reflected in the function, logic, and interface of the applications.

- Context level

The evaluation in the context level concentrates on the influencing factors that lie in the context acquisition, management, and utilisation. The relevant parameters are efficiency of context abstraction, efficiency and interoperability of context distribution, and sharing and understanding of context. Performance at this level may play an important role in context-aware mobile applications.

- Physical level

The physical level is about the real-world factors influencing the usability and user satisfaction of context-aware mobile systems. The main parameters are unobtrusiveness of context acquisition, speed and accuracy of context acquisition, quality of devices (embedded and mobile), and quality of wireless network. This level is under the application and service level, but it plays an important role as parameters in this level may affect performance of the upper levels of the applications.

This evaluation framework is based on the analysis of the specificity of context-aware mobile systems, which takes the issues relevant to use of context seriously. As the use of context may bring risks to the efficiency and security of the systems which may directly influence the performance of the context-aware mobile system, it is necessary to give enough consideration of the context related aspects in the evaluation.

7.3 Summary

This chapter focuses on the evaluation of context-aware mobile systems and provides an evaluation framework with three levels for context-aware mobile systems. In this chapter, the characteristics of mobile device-based mobile applications are identified, and then the context-aware mobile applications are examined with respect to the context acquisition and context computation. The evaluation framework is proposed according to the characteristics and specificities of the context-aware mobile computing systems.

Chapter 8 Case Studies on Context-aware Automation and Decision Support

Following the design science research methodology, Chapters 4, 5, and 6 have clarified the key techniques, models, methods that constitute the context-aware computing architecture. In order to demonstrate the feasibility of the presented design and find the answer to the research questions, two case studies are elaborated in this chapter. Both case studies have employed the context of computation tasks to minimise users' efforts in the application with the solutions proposed in this investigation. Results demonstrate the feasibility of the proposed methods and show great potential in utilising context information as implicit input to facilitate users' operations, and some potential problems are also indicated.

8.1 The Case Studies – A Brief Introduction

The purpose of the case studies is to demonstrate the feasibility of the design and verify the effectiveness of the methods proposed in this investigation in providing a context-aware computing service. Two case studies will be elaborated with experimental studies – MES user authentication and Smart Home System (SHS).

(1) MES user authentication – Context-aware automation

The DONE-IT MES is a European Commission funded project which aims to employ student oriented Internet enabled mobile techniques for classroom examinations. Students use their mobile devices to login the system, load the questions, and then answer the questions on mobile devices. Because of the resource limitation of the mobile devices, it is challenging to guarantee the usability and efficiency in user operation of the mobile application and minimise the congestion in data communication for concurrent use by numerous users. Therefore, the context-aware paradigm is involved, to make use of the mobile built-in sensing techniques to efficiently utilise the limited resource and enhance the efficiency of user interaction with the proposed scheme in this investigation.

The MES user authentication strategy is introduced as one of the two case studies. Users' student ID card barcode, NFC data, mobile device ID, time, and location, etc. are used as context to facilitate the authentication task. The built-in sensing techniques such as touch screen, camera, NFC module are employed for user authentication, and the accelerometer-based user activity recognition is used for quick operations. Results show that the methods based on the novel sensing techniques can effectively enhance the user interaction in the login operation. Quantitative evaluation is provided based on experimental studies. Part of the work about MES authentication presented in section 8-2 is published in the list of publications [4].

(2) Smart Home System – Context-aware decision support

In the smart home project, the mobile built-in and embedded sensors are employed for context-aware automatic control of home facilities with the proposed methods. The mobile built-in sensors are used predominantly to characterise the users, and the embedded sensors are used to describe the environment context in the home environment. The system makes use of the context information to predict users' expectations and provides an appropriate computing service for home facility control proactively and automatically.

In addition to context-aware automation, this project exploits the history context with attribute reduction and rule generation methods for the supervision of computing service adaptation. The design, methods, and techniques are presented, quantitative and qualitative evaluations are illustrated to justify the effectiveness of the proposed service customisation and rule generation methods. Results show that the methods are feasible for context-aware application scenarios, and the context-aware paradigm is promising to facilitate people's daily life in their home environment.

The case studies are elaborated in the following sections, implementing the designs and methods presented in Chapter 4, 5, and 6.

8.2 Case 1: MES User Authentication with Built-in Sensing Techniques

The traditional method of user login authentication of an application is through username and password manual input with a keyboard. However, the user authentication of a mobile application is usually inconvenient because of the limited size of a hard keypad or touch screen, which makes the user operation time consuming and error prone. Therefore, the mobile built-in sensing techniques are employed to enhance user interaction. Through this case study, the following objectives are expected to be achieved:

(1) To demonstrate the feasibility of the context-aware methods based on the built-in sensors in improving the usability and efficiency of mobile applications,
(2) To identify how the built-in sensing technique can enhance the user interaction of mobile applications, and
(3) To determine the influencing factors affecting the efficiency of the user interaction with the sensing techniques.

This case study focuses on the context-aware service customisation in improving the user interaction of mobile applications with the proposed context-aware architecture and methods.

8.2.1 Techniques Employed

To improve the usability and efficiency of the mobile application, the first step is to find applicable and promising technical approaches. Normally, each student has a unique ID card with a barcode printed on the surface and a built-in NFC chip inside. The camera-based image processing and NFC recognition techniques can be used for user identification and authentication to simplify the user operation and improve the operational speed and accuracy. In addition, the accelerometer can be used to assist the user interaction with mobile devices, which also eases their operation. The microphone-based speech recognition is technically feasible, but it is not appropriate for the silent environment and strict security requirement of an examination.

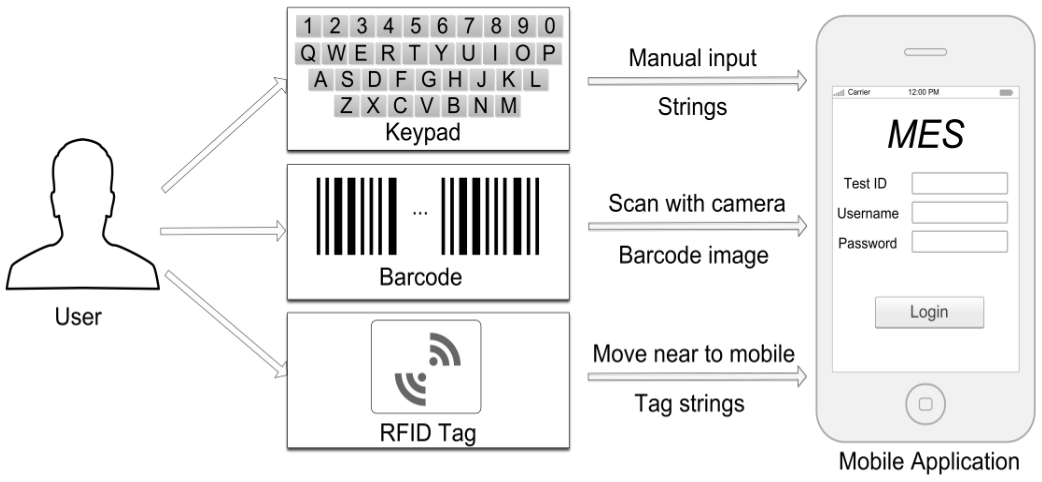


Figure 8-1 Technical Approaches Employed for MES User Authentication

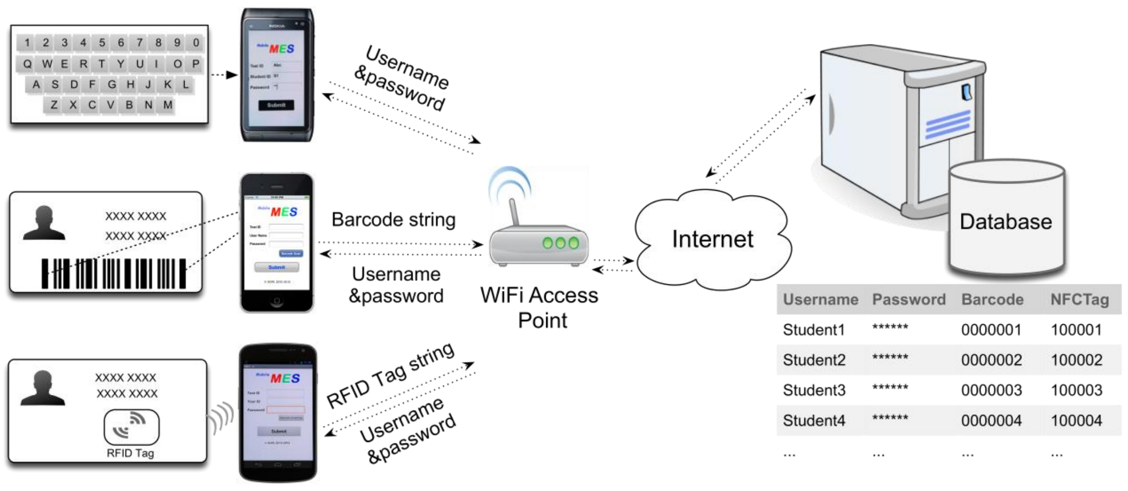
As shown in Figure 8-1, besides touch screen manual input, two technique approaches - student ID card barcode scanning and student ID card NFC recognition - are exploited to improve the efficiency and usability of MES user authentication, and the methods are evaluated with the latest mobile platforms. In the following sections, the design and implementation, testing, and evaluation of the interaction techniques with the proposed context-aware scheme are presented to improve the usability and operational efficiency.

There are many attributes that can be used to assess the usability and efficiency of the sensing technologies in user interaction such as whether they are easy to use or not, operation speed, error rate, vulnerable to interference or not, computational complexity, arbitrary degree of input data, and field of applications (Hornbæk, 2006). The evaluation of the user interaction methods focuses on two methods: comparing performance of the different methods on the same platform, and determining how the performance of techniques varies on the platforms.

8.2.2 Design and Implementation

8.2.2.1 Design of MES user authentication module

As the MES system needs to distribute questions to students on mobile devices and collects students' answers in real-time, the distributed client-server architecture is employed. Students use their mobile clients to access the questions and then answer the questions by operating on the mobile devices. The centralised server stores all the data in a database and then marks the answers instantly. For cross-platform use, the mobile client application should support the mainstream mobile operating systems. System architecture of MES is shown in Figure 8-2.



Username	Password	Barcode	NFCTag
Student1	******	0000001	100001
Student2	******	0000002	100002
Student3	******	0000003	100003
Student4	******	0000004	100004
...	...	...	...

Figure 8-2 System Architecture of MES System

For user authentication, the username and password of the authorised users are stored in the database with the corresponding barcode stings and NFC tags. When a user logs in with barcode or NFC magnetic card, the mobile device gathers the data from the barcode and NFC first and then retrieves the corresponding username and password from the remote server. The username is then displayed while the password is hidden (shown as '*') on the interface. User can confirm the ID to login to the system, or he/she can try again if a problem occurs. With camera and NFC module, users can login to the system with student card conveniently.

8.2.2.2 Context Items and Rules Defined for MES Authentication

For MES login, the user account (username and password) and examination time are used for user authentication. In order to guarantee the security of the system and keep malicious access from the system, some other attributes about the users and device, such as barcode

NFC tag, International Mobile Equipment Identity (IMEI) number, phone number, location, and time are involved as context information to assist the authentication. The information selected as context and their expression are given in Table 8-1.

Table 8-1 Context Items Used in MES

Source	Context Item	Denoted By
Camera	Barcode	C_barcode
NFC module	NFC Tag	C_nfctag
Smartphone	IMEI	C_imei
Smartphone	Phone number	C_phoneno
GPS module	Location	C_location
Smartphone	Time	C_time
Touch screen	User account	C_account

According to the graphical representation of context model presented in Figure 5-3, the context model tailored to the MES authentication use case is as shown in Figure 8-3.

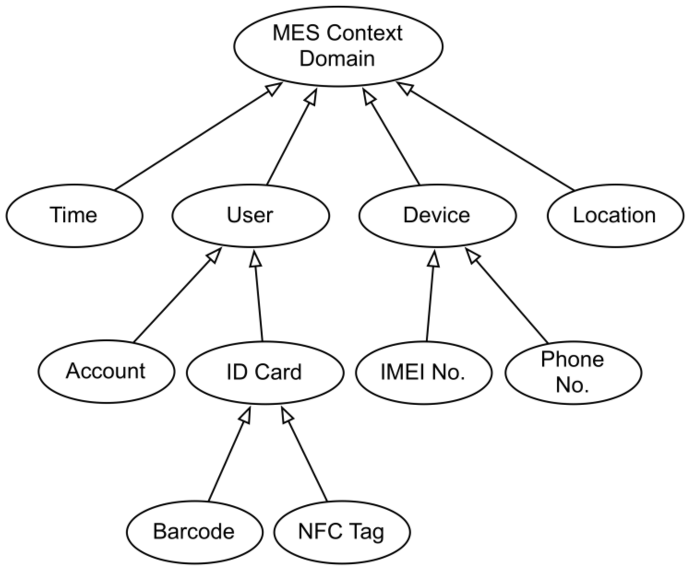


Figure 8-3 Context Model for MES User Authentication

As shown in Figure 8-3, the involved context information can be provided with the built-in modules of the mobile device. Since IMEI number and phone number are both unique to the users, only the IMEI number is used to identify users' device in MES system. Then, the rules for the context-based authentication are defined in Table 8-2.

Table 8-2 Rules for MES User Authentication

ID	Rules (Conditions and Decisions Attributes)
Rule1	($C_{barcode}$ is True or C_{nfctag} is True)∧ (C_{imei} is True) ∧ (C_{time} is True) ∧ ($C_{location}$ is True) ⇒ Login is Authorised
Rule2	($C_{barcode}$ is False or C_{nfctag} is False)∧ (C_{imei} is ?) ∧ (C_{time} is ?) ∧ ($C_{location}$ is ?) ⇒ Login is Denied
Rule3	($C_{barcode}$ is True or C_{nfctag} is True) ∧ (C_{imei} is False) ∧ (C_{time} is True) ∧ ($C_{location}$ is True) ⇒ Input Password
Rule4	($C_{barcode}$ is ? or C_{nfctag} is ?)∧ (C_{imei} is ?) ∧ (C_{time} is False) ∧ ($C_{location}$ is ?) ⇒ Login is Denied
Rule5	($C_{barcode}$ is ? or C_{nfctag} is ?)∧ (C_{imei} is ?) ∧ (C_{time} is ?) ∧ ($C_{location}$ is False) ⇒ Login is Denied

In the rules, username and password login is regarded as the most trustworthy way. For Barcode scanning and NFC recognition, the authentication is evaluated with IMEI, time, and location. If the user logs in with an authorised barcode or NFC tag at examination time and he/she is in the exam area, login is authorised. But if it is not at exam time or the user is not in the exam area, login is denied. On the other hand, if the user logs in with an authorised barcode or NFC tag but not with the user's commonly used IMEI number, the user is required to input his/her password for further authentication. With the rules and context information, the Barcode scanning and NFC recognition become more trustworthy. They can identify malicious access to the system with other users' ID card or device and remote login for other purposes.

8.2.2.3 Implementation

Table 8-3 Development Kits for MES Mobile Client Applications

Platform	Development Kit	Version	Programming Language
iOS	iOS SDK	4.3	Objective-C
Android	Android SDK	20.0.1	Java & XML
	JDK	6	
	Eclipse	3.7.1	
WP7	MS Visual Studio Express for Windows Phone	2010	C# & XAML
Symbian	Qt	4.7.4	C ++

Implementation of the system consists of a lot of work including mobile client application, image processing, NFC recognition, and service end application. Students may hold mobile devices of different models. Therefore, this system is expected to cover the mainstream mobile operating systems, i.e. Apple iOS, Google Android, Microsoft Windows Phone 7, and Nokia Symbian. The platforms, development kits (version), and programming languages for mobile application development are listed in Table 8-3.

(1) Design with the Hierarchical System Architecture

a) Physical layer

The physical layer deals with the acquisition of context with the smartphones.

- *Mobile built-in sensor acquisition* - The context barcode, NFC tag, IMEI number, phone number, location, and time are obtained with the smartphone devices.
- *User input* - The user account are obtained through user manual operations on the touch screen.

b) Context handling layer

Context handling layer are responsible for context management and context distribution.

- *Context integration* - The context values obtained are abstracted and normalised into a structured format.
- *Context update* – Context update is done by smartphone with HTTP request.
- *Context query* – Interfaces are provided for service applications and mobile clients to access context data.

c) Context-aware service layer

- *Context-aware service customisation* - The service customisation is accomplished with semantic distance based rule matching.
- *Context-aware APIs* – The example context-aware service interfaces for context update and context request are:
  contextUpdate?DATA=
  contextQuery?barcode=?
- *Mobile client applications* – The mobile client application provides interfaces for user operations, such as barcode scanning, NFC recognition, and manual input.

This case study focuses on the demonstration of the feasibility to integrate the smartphone built-in sensing techniques to improve the performance of mobile applications with the proposed system architecture and service customisation method. The rule generation layer which is used to supervise the service customisation is not implemented in the current stage.

(2) Rule-based Decision Making

The user authentication decision making is completed according to the context model in Figure 8-3 and rules defined in Table 8-2 using the method presented in 6.3.1.

The condition attributes in the MES system are:

$$A=\{C_{barcode}, C_{nfctag}, C_{imei}, C_{time}, C_{location}\}$$

The decision attributes for home facility control:

$$D=\{d_{authorised}, d_{password}, d_{denied}\}$$

is decided according to the semantic distance between the context sample and the rules.

In order to demonstrate how decision making works in MES authentication, a set of context data in database of the prototype system is selected as the current context:

$$CV_i=\{True, ?, False, True, True\}$$

The *dist(CV_i, Rulei)* denotes the semantic distance between the context sample CV_i and rule *Rulei*. According to Figure 8-3, the barcode in the tree hierarchy indexing structure in the context model can be represented with the diagrams in Figure 8-4.

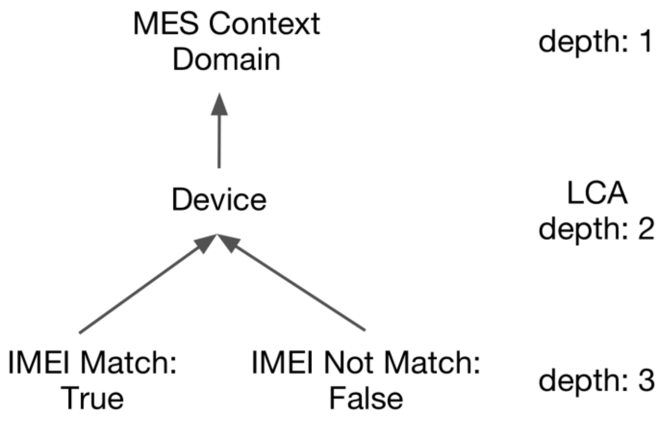


Figure 8-4 Tree Hierarchy Structure of Barcode in the MES Context Domain

Then, according to rule matching presented in section 6.3, the normalised semantic distance between the context sample CV_i and the rules in Table 8-2 can be calculated.

For example, in the sample data $CV_i[C_{imei}] = False$. According the tree hierarchy structure, *depth(LCA(IMEI Match:True, IMEI Not Match:False))* = 2, *depth(IMEI Match:True)* = 3, *depth(IMEI Not Match:False)* = 3. According to formula (6-8) and (6-9),

$$\mathrm{dist}(CV_i[C_{imei}], Rule1[C_{imei}]) = 1 - 2*2/(3+3) = 0.33$$

With the same method:

$$\mathrm{dist}(CV_i[C_{barcode}], Rule1[C_{barcode}]) = 1 - 2*4/(4+4) = 0$$

Then, accordint to (6-10),

$$\mathrm{dist}(CV_i, Rule1) = 0.25*(\mathrm{dist}(CV_i[C_{barcode}], Rule1[C_{barcode}]) + \mathrm{dist}(CV_i[C_{imei}], Rule1[C_{imei}]) + \mathrm{dist}(CV_i[C_{time}], Rule1[C_{time}]) + \mathrm{dist}(CV_i[C_{location}], Rule1[C_{location}])) = 0.0825$$

where 0.25 is the weight of the four relevant context items.

So, the results of the rule matching are given as follows:

(1) $dist(CV_i,Rule1) = 0.25*(0+0.33+0+0)=0.0825$
(2) $dist(CV_i,Rule2) = 1*0.25=0.25$
(3) $dist(CV_i,Rule3) = 0.25*(0+0+0+0)=0$
(4) $dist(CV_i,Rule4) = 1*0. 5=0.5$
(5) $dist(CV_i,Rule5) = 1*0.5=0.5$

In this system, the weight of the attributes in the rules are equal, the weight is *1/n* when there are *n* attributes in a rule. The semantic distance between *CVi* and the rules are presented in Figure 8-5.

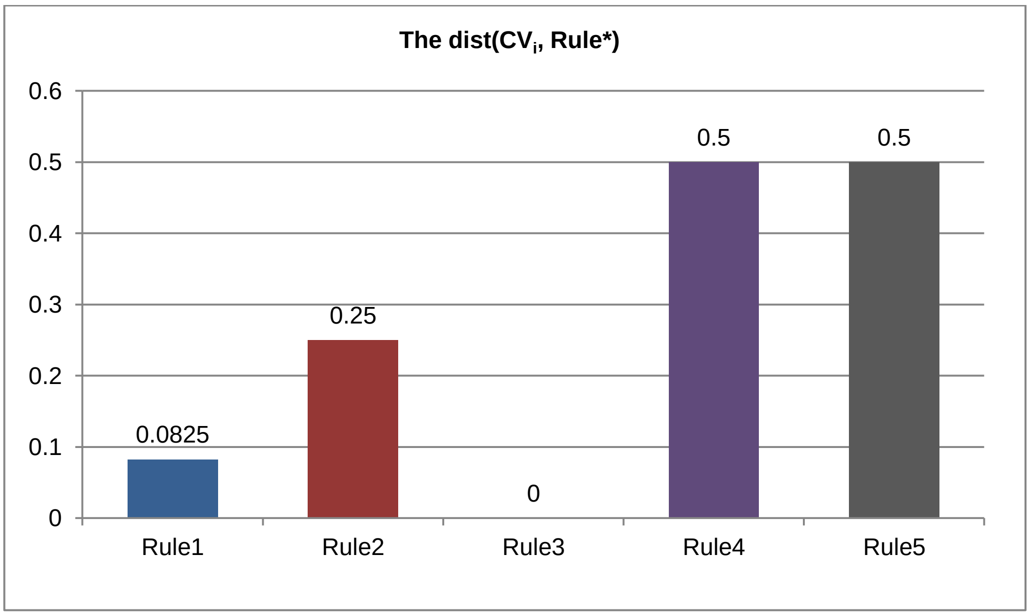


Figure 8-5 Semantic Distance between CV_i and the Rules in MES User Authentication

It is clear that the minimum semantic distance between CV_i and the pre-defined rules is *dist*(CV_i,*Rule3*). Thus, *Rule3* is considered to be the appropriate rule for user authentication, and the decision '$d_{password}$' is executed, namely password is required.

8.2.3 Testing and Evaluation

This section provides the experimental studies to test and evaluate the operations, speed, and error rate of the three user interaction methods and identify how the two sensing technique-based methods outperform the manual input method. Evaluations are conducted to evaluate performance of the different methods and performance of the methods on different platforms. In the following, the operations are evaluated first, and then the efficiency and error rate.

8.2.3.1 The Use Operation is Simplified

(1) Barcode scanning method

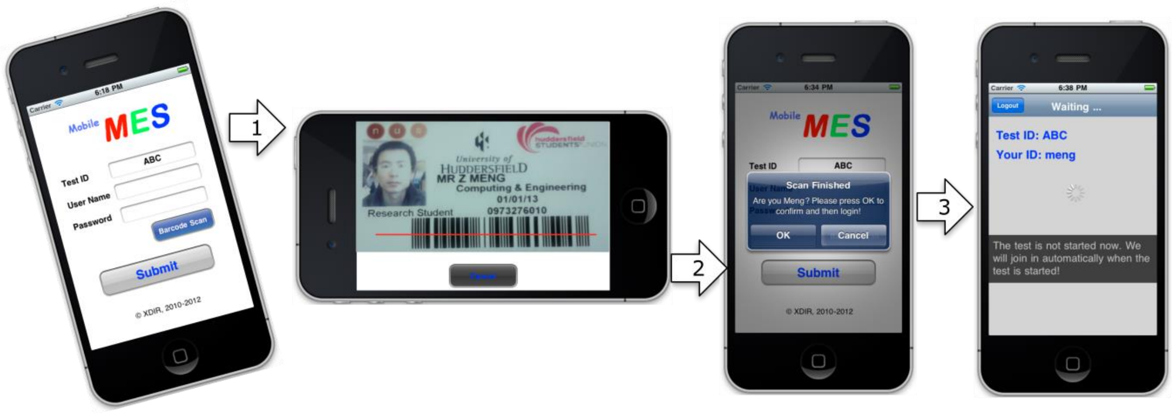


Figure 8-6 Work Flow of Barcode Scanning Method on iOS Platform

The barcode scanning method has largely facilitated user operation with the application of the built-in camera on the mobile devices. The operation steps are: Firstly, launch the MES application on mobile devices and input the Exam ID. Then, follow steps 1, 2, and 3 as shown in Figure 8-6:

1) Shake the phone or press "Barcode scan" button to initiate the camera,
2) Target the red line in camera over barcode till it finishes (Beep when it finishes),
3) Confirm matched ID to login.

The user just needs to shake the phone or press a scanning button, target the barcode in the camera vision, and confirm to login. It is easier to operate compared with the operations using a soft keypad on size limited touch screens.

(2) NFC recognition method

The NFC recognition also largely simplifies the operations of the user operations in the login process. The operation steps of the NFC recognition login with a student ID card are: Launch the MES application on mobile devices and input the Test ID. Then, follow steps 1 and 2 in Figure 8-7:

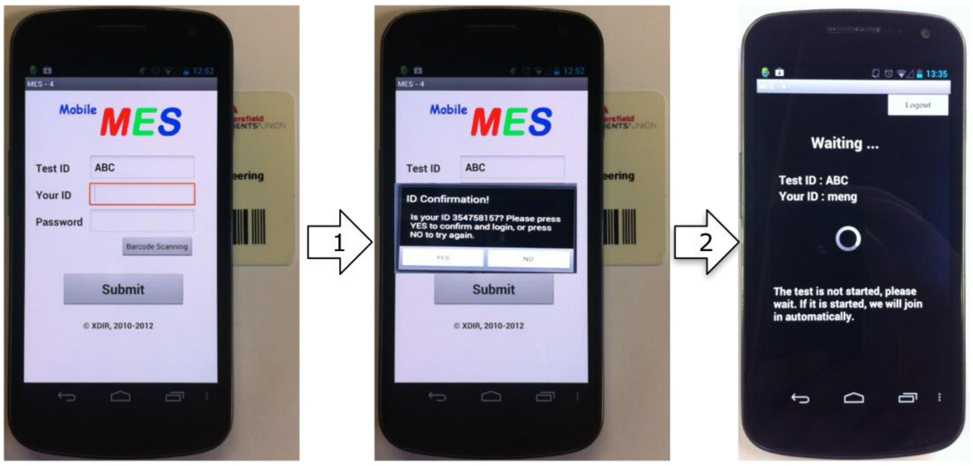

MES - 4
Mobile MES
Test ID
ABC
Your ID
Password
Barcode Scanning
Submit
© XDIR, 2010-2012
1
ID Confirmation!
Is your ID 354758157? Please press YES to confirm and login, or press NO to try again.
YES
NO
2
Logout
Waiting ...
Test ID : ABC
Your ID : meng
The test is not started, please wait. If it is started, we will join in automatically.

Figure 8-7 Work Flow of NFC Recognition Method

1) Move the card close to the device until recognition is finished (Beep when it finishes),
2) Confirm matched ID to login.

The NFC recognition method converts the operations of typing characters on a soft keyboard to moving a student ID card close to the device, which is easy and convenient to operate.

In order to determine how much they can improve the operational efficiency, the experiments are carried out and quantitative evaluation is conducted.

8.2.3.2 Evaluation of Operational Efficiency

In order to obtain the objective data to demonstrate the advantage of the novel hardware techniques in user operation, the testing condition, equipment, and strategy should be specified in advance. The experimental studies strive to find the answers to the following questions:

1) How much do the applied technologies enhance the efficiency of mobile applications?
2) How do they differ among the different mobile hardware platforms?
3) What are the influencing factors of these technologies?
4) What are the strengths and weaknesses of these applied technologies?

With these questions, the methods implemented in the system, including the touch screen manual input, barcode scanning, and NFC recognition, are tested separately. The parameters to be measured are speed (operation time) and error rate, and the mean value and standard deviation (short for STDEV) of operation time can be calculated for further analysis. Mean value of sample data can be used to estimate the real value of parameters, and standard deviation is a parameter that can be used to describe the degree of scatter of data. Thus, they can be used to evaluate the efficiency and stability of the methods in user interaction. The strategies of the test and evaluation are:

1) Test the different methods on the same platform with the same condition to compare the different interaction techniques,
2) Test the same methods on different mobile platforms with the same condition to examine its performance on different mobile platforms,
3) Determine the underlying influencing factors of these methods based on the experiment data.

As some methods, such as the barcode scanning, are easily affected by the ambient environment such as illumination brightness and light reflection, the testing of different devices are expected to be done in identical conditions and environment. For the different approaches, the accounts to be used are randomly generated by a specially designed algorithm. Each method is repeatedly tested 20 times on different mobile devices. The lengths of username and password are set from 8 to 20 alphabets in even values. The touch screen manual input, barcode scanning, and NFC recognition are tested separately with the same user. The time is gathered using the timer function provided with the mobile platforms, and the accuracy is 1 millisecond. The starts and ends of timing in the tests are set as below:

- Touch screen input: From launching the application to completion of manual input
- Barcode scanning: From shaking/pressing barcode scan to when barcode is shown and confirmation is required
- NFC recognition: From launching the application to when tag string is shown and confirmation is required

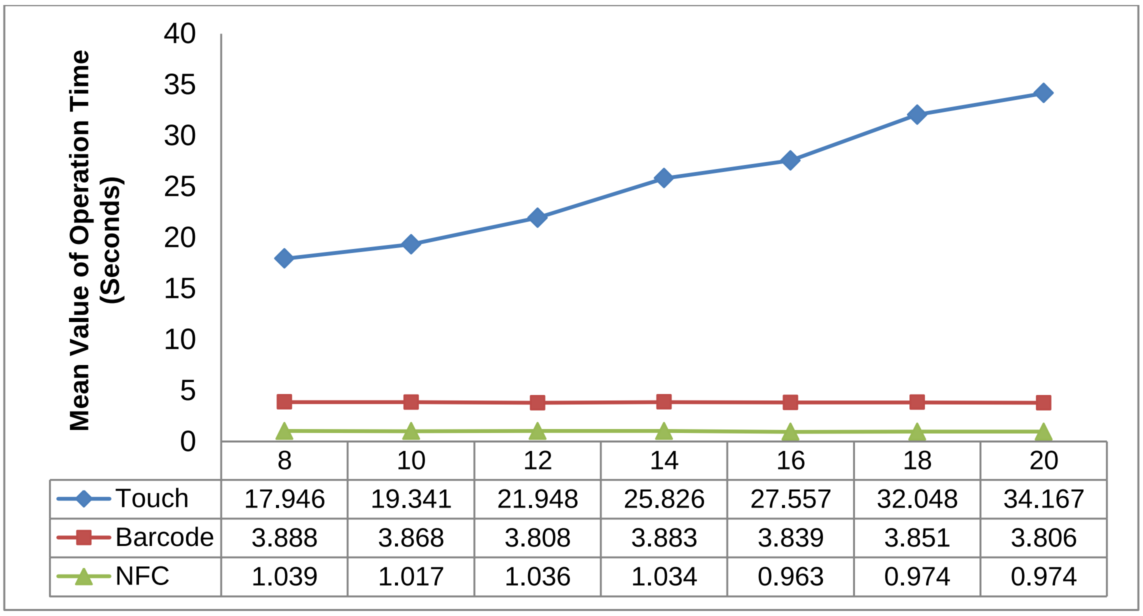
Table 8-4 Devices Used for Evaluation

Platform	Device	OS Version	Camera (Pixels)	Screen (Inches)
iOS	Apple iPhone 4S	iOS 4.3.5	5M	3.5
Android	Samsung Galaxy Nexus S	Android 4.1.1	5M	4.65
WP7	HTC HD2	Windows Phone 7.1	5M	4.3
Symbian	Nokia N8	Symbian^3	12M	3.5

The mobile devices employed in the testing with their mobile OS, OS version, camera resolution, and screen size which are relevant to user interaction are listed in Table 8-4.

(1) Performance of the context-aware applications on the same platform

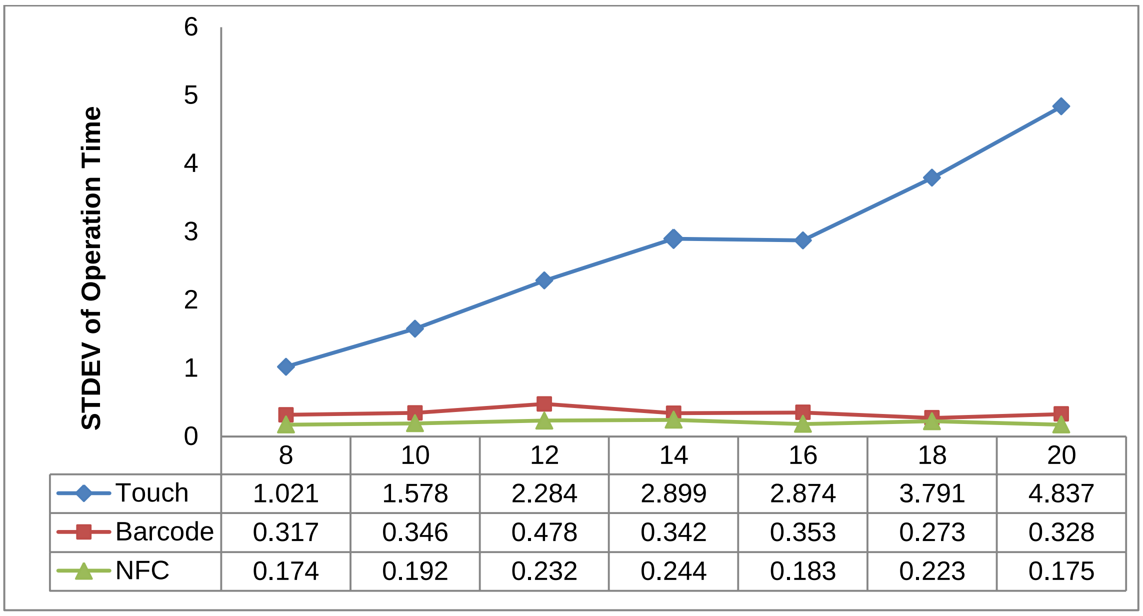
For login with the three interaction techniques, the mean values and standard deviations of operation time on Android device Galaxy Nexus S are given in Figures 8-8 and 8-9. Since NFC technique is still not widely shipped on the smartphones, the three interaction methods on one platform are achieved on the Android platform only and it is tested with the Nexus S. From Figure 8-8, it is not difficult to find that the NFC approach is the fastest (Average[MEAN, STDEV]=[1.005, 0.203]), barcode scanning is medium (Average[MEAN, STDEV]= [3.849, 0.348]), and the touch screen manual input is the slowest (Average[MEAN, STDEV]=[25.548, 2.755]). Apparently, the NFC recognition approach and Barcode scanning methods cost only 3.93% and 15.07% operation time of the commonly used touch screen manual input method. Thus, the barcode and NFC technique is more efficient than touch screen for user interaction.



	8	10	12	14	16	18	20
Touch	17.946	19.341	21.948	25.826	27.557	32.048	34.167
Barcode	3.888	3.868	3.808	3.883	3.839	3.851	3.806
NFC	1.039	1.017	1.036	1.034	0.963	0.974	0.974

Figure 8-8 Mean Value of Operation Time with the 3 Methods on Nexus S

From the curve in Figure 8-8, it is clear that the operation time using the barcode scanning and NFC recognition is obviously shorter than the touch screen. In addition, it is easy to find that the operation time in the interaction regularly increases with the length of information input with the touch screen manual input method, namely the longer the information to input, the more time it costs. It remains consistent utilising either the barcode scanning or NFC recognition approach as the same amount of work is done, no matter how long the username and password are.



	8	10	12	14	16	18	20
Touch	1.021	1.578	2.284	2.899	2.874	3.791	4.837
Barcode	0.317	0.346	0.478	0.342	0.353	0.273	0.328
NFC	0.174	0.192	0.232	0.244	0.183	0.223	0.175

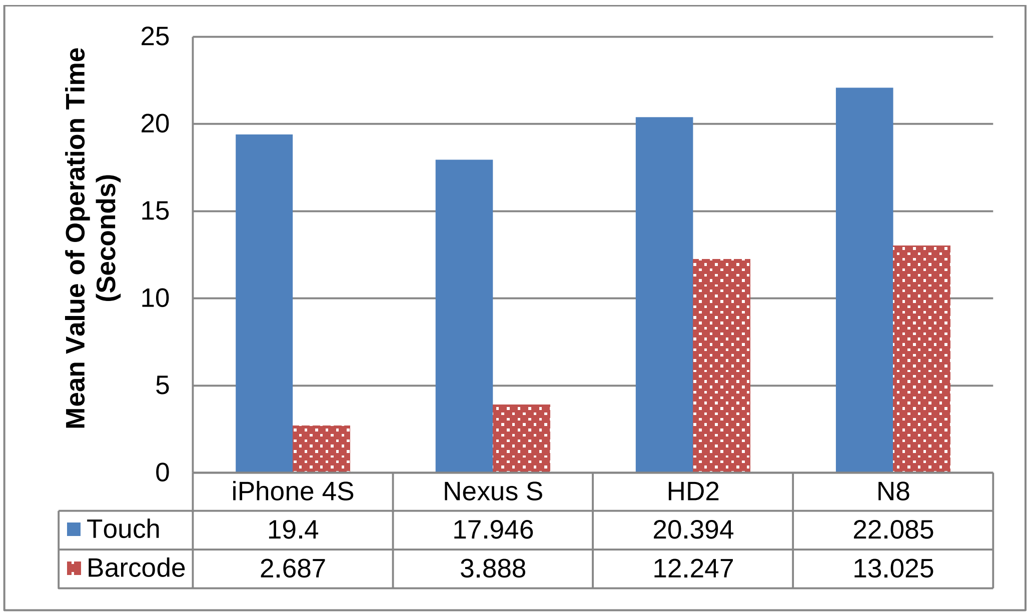
Figure 8-9 Operation Time STDEV with the 3 Methods on Nexus S

Then, from Figure 8-9, the standard deviations of the operation time vary in different trends as well. The standard deviation of touch screen manual input increases with the length of the information to input. That means the longer the information to input, the more uncertainty there will be over the operation time of the input, while the standard deviation of barcode scanning and NFC recognition methods remain small and smooth. Namely, no matter how long the data to input is, the operation time of the interaction is theoretically constant. That is to say, the barcode and NFC methods are more stable in interaction compared with the touch screen.

(2) Performance of the context-aware applications on different platforms

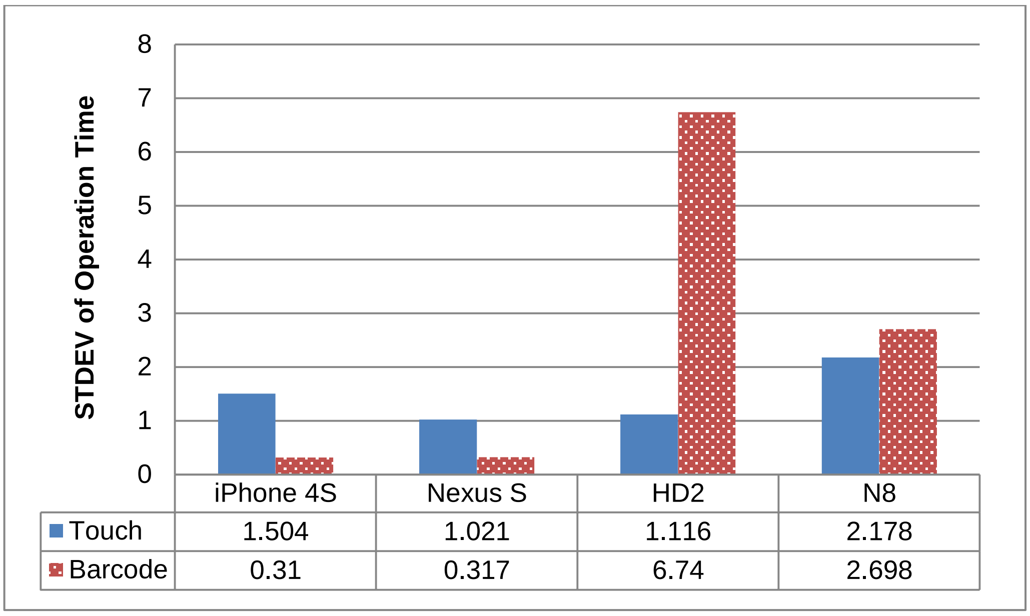
The touch screen manual input and barcode scanning on the four mobile platforms are also tested with user name and password length of 8 alphabets that is randomly generated, The environmental condition of the testing is considered to guarantee that the external effect will not become a critical problem influencing the performance of the methods. NFC is not tested across platforms simply because the NFC is not widely shipped on mobile devices, but is limited to several latest models.

With the testing results, the parameters mean value and STDEV are calculated to evaluate the efficiency and stability. The mean value, standard deviation, and error rate of the two methods - barcode scanning and NFC recognition on the four platforms are presented in diagrams in Figures 8-10 to 8-12.



	iPhone 4S	Nexus S	HD2	N8
🟦 Touch	19.4	17.946	20.394	22.085
🟥 Barcode	2.687	3.888	12.247	13.025

Figure 8-10 Operation Time Mean Value of Touch Screen and Barcode Scanning



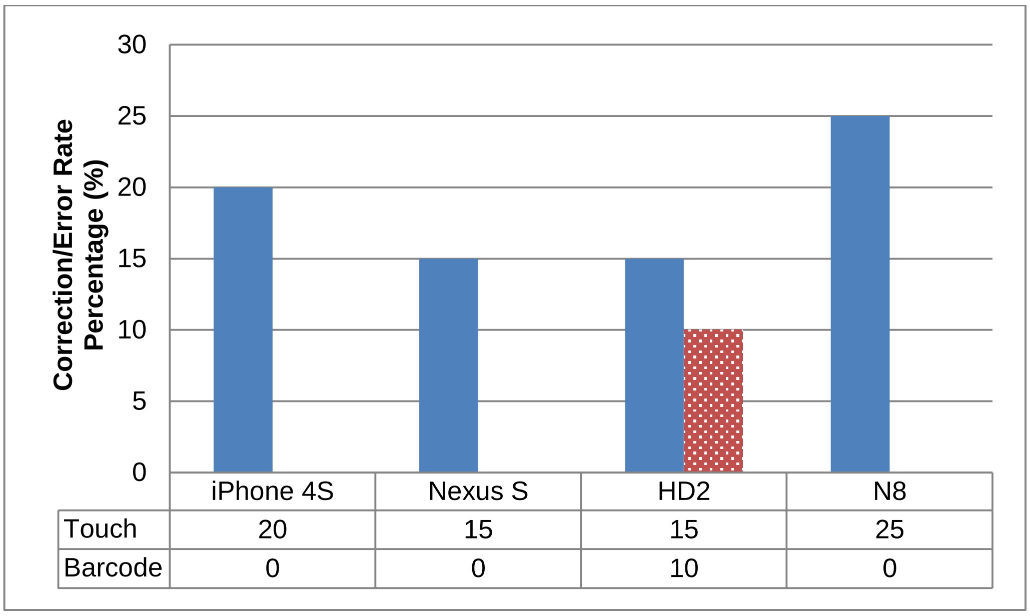
	iPhone 4S	Nexus S	HD2	N8
🟦 Touch	1.504	1.021	1.116	2.178
🟥 Barcode	0.31	0.317	6.74	2.698

Figure 8-11 Operation Time STDEV of Touch Screen and Barcode Scanning

Figure 8-10 shows the mean operation time in the user authentication of the two approaches - touch screen manual input and barcode scanning. It reveals that the barcode scanning method is faster than the touch screen manual input, especially on iPhone 4S (MEAN=2.687s) and Galaxy Nexus S (MEAN=3.888s). Moreover, the operation speed of either method varies on the different mobile platforms. For the manual input method, the operation on Android device Galaxy Nexus S (MEAN=17.946s) is the fastest because of the biggest screen size (4.65

inch). Conversely, the operation on the Symbian^3 devices Nokia N8 (MEAN=22.085) is the slowest due to the smallest screen size (3.5 inch) and the interface not being as convenient as that of the iPhone 4S (MEAN=19.400s) with the same screen size. For the barcode scanning approach, the operation speed varies more remarkably on different mobile platforms. The reason may lie in the control efficiency of the camera and the quality of camera lens. The iPhone 4S is faster, as the control of camera is very smooth. However, the barcode scanning on HTC HD2 (MEAN=12.247) and Nokia N8 (MEAN=13.025) are much slower; this is because the control of the camera is not efficient enough, and the lens adjusts again and again.

Figure 8-11 shows the standard deviation of operation time of these two methods in the user authentication. The results illustrate that the standard deviation of the manual input on different devices is largely consistent. That is because the manual input is not easily affected by some objective conditions. Conversely, the barcode scanning method is high in standard deviation except the iPhone 4S (STDEV=0.310) and Android (STDEV=0.317) as the cameras on iPhone 4S and Nexus S are of high quality and focus fast. The Nokia N8 (STDEV=2.698) is high in standard deviation because the control of the camera is not very efficient, and the HTC HD2 (STDEV=6.74) is also high because the camera zooms and focuses very slowly. In addition to the focus speed of the camera, the quality of image is also an influencing factor.



	iPhone 4S	Nexus S	HD2	N8
Touch	20	15	15	25
Barcode	0	0	10	0

Figure 8-12 CR of Touch Screen and ER of Barcode Scanning

Finally, Figure 8-12 illustrates the Correction Rate (CR) of the manual input method and Error Rate (ER) of the barcode scanning method. The correction rate is denoted with the percentage of the logins that are with error operations which are corrected by user. The Galaxy Nexus S (CR=15%) and HTC HD2 (CR=15%) are lower in correction because the screen sizes are bigger (4.65 inch and 4.3 inch) than the others. However, Nokia N8 (CR=25%) is the highest in correction rate because its keys on the soft keypad are too small to operate. On the other hand, the error rate of HTC HD2 (ER=10%) is high because the camera lens is susceptible to light illumination and the image quality is usually poor. The error rates of barcode scanning on other devices are low because the cameras can easily obtain high quality images.

8.2.4 Summary

This case study adopts the smartphone built-in sensing techniques for user authentication with the proposed context-aware system architecture and service customisation strategy. The user interaction is simplified and usability and efficiency of the mobile application are promoted with the camera-based barcode scanning and NFC recognition methods

The methods presented in this work are successfully implemented. Through the testing and evaluation of the system, it is found that the utilisation of the novel sensing techniques with the context-aware methods can effectively relieve the inherent problem in operational efficiency of touch screen manual input due to limited screen size. The utilisation of these novel built-in modules converts the manual keypad input into shaking the phone, targeting an object in camera, and moving a card close to the mobile device. Thus, the barcode scanning and NFC recognition techniques have simplified the operations of the MES login process. On the other hand, both methods can largely promote operation speed and reduce error rate in the operations compared with the traditional touch screen manual input method. The usability and efficiency of the user interaction of mobile application is therefore raised. Consequently, it can be concluded that it is feasible to employ the built-in sensing techniques to enhance the usability and efficiency of the mobile applications with the proposed methods.

However, the performance of context-aware systems based on the sensing technologies may differ in speed and accuracy among different mobile platforms. In other words, the performance of the hardware-related interaction approaches is dependent on the capability of the built-in hardware modules. Since the heterogeneity problem of mobile devices objectively exists, the extent to which the selected sensing techniques can promote the performance of mobile applications is dependent on the coordination of hardware and related algorithms.

In addition, it is found in the testing that different hardware or sensing technologies may be affected by different environmental conditions. The technologies may make sense only when the interference is acceptable. For instance, the barcode scanning method can be used only when the ambient illumination is appropriate and light reflection is acceptable, and NFC recognition method can be used only with devices shipping the NFC module.

8.3 Case 2: SHS Context-aware Automation and Decision Support

This case study employs the mobile devices and embedded sensing techniques for home facility control with the context-aware methods proposed in this investigation. In order to provide appropriate computing service that is customised to the contextual situations, the various built-in sensors in smartphones and some separate sensors are used to perceive the user's ambient context. The embedded electronics are used to interface the separate sensors and actuators to the computer system.

As a case study, the SHS system aims to use the novel sensing techniques to assist human decisions for home facility control by predicting the user's intent. The following objectives are expected to be achieved in the design and implementation:

(1) To demonstrate the feasibility of the design and implementation with context-aware architecture, methods, and theories presented in this investigation,
(2) To evaluate the rule-based context-aware service customisation and rule generation method for proactive computing service provisioning, and
(3) To prove the competence and usefulness of the mobile techniques and sensing techniques for decision assistance in practical use of SHS environment.

In order to achieve the above objectives and effectively evaluate the proposed methods, the system design, implementation, results and evaluation are presented in the following sections.

8.3.1 System Design and Implementation

The main work to do in achieving the SHS system are: (1) gathering context data with heterogeneous sensing techniques, (2) context abstraction and context reasoning, (3) context-aware service customisation, and (4) rule generation based on history context for high-level supervision. The design and implementation work is provided to achieve these main tasks.

8.3.1.1 SHS System Architecture Design

The design and implementation of a smart home system requires knowledge in the fields of sensor networks, embedded electronics, wireless communication, mobile computing, and computer science. It integrates a lot of devices and technologies which makes the design and implementation more complicated than for a traditional computing service system. Generally, devices and connections in the system can be presented with Figure 8-13. The sensor and actuators with wireless transceivers are installed in the home area, and the context information is collected and stored in the service end for context-aware automation. The system is designed following the guidelines of the hierarchical system architecture presented in Chapter 4, which divides the systems into four layers: physical layer, context handling layer, context-aware service layer, and rule generation layer.

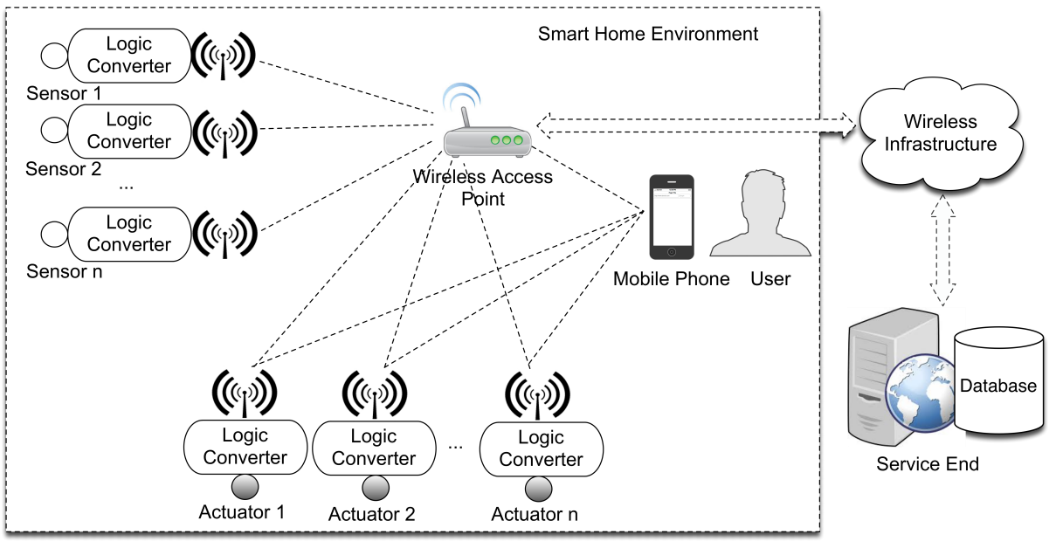


Figure 8-13 Devices and Connections in Smart Home System

(1) Physical layer

The physical layer consists of the hardware devices employed in the home area including sensors, actuators, embedded electronics, wireless transceivers, and the wireless infrastructure. The main functions of the physical layer are listed as below:

a) *Context acquisition* - The sensors gather the context information under the control of the embedded controller such as a MCU or DSP processor. Normally, the raw sensor data needs pre-processing to improve the quality of the context data.
b) *Context data transmission* - In order to make the context data available to the service end, the embedded controllers are interfaced to the wireless infrastructure through a wireless transceiver, such as RS-232 to WiFi transceiver RN-370M. The setting of RN-370M is given in Appendix 2.

c) *Command execution* - The physical layer can also receive and parse the context-aware service from the service end and execute the command automatically. This is accomplished by the MCU controller and actuators.

The functions of physical layer are essential to the smart home system, as availability and quality of context, efficiency of data transmission, and stability of command execution may largely influence the performance of the whole system.

(2) Context handling layer

The context handling layer consists of the functional modules for context management and context distribution, including:

a) *Context integration* - The sensor values are abstracted and normalised into a structured format that is convenient for communication and computation.
b) *Context reasoning* - High-level contexts are derived from the low-level context with context reasoning model.
c) *Context update* - Context update is done by the embedded clients with HTTP requests.
d) *Context query* - Context query provides interfaces for service applications and mobile clients to access context data.

This layer integrates the separate sensor values to context data that is applicable to the system, and formalise the data to structured formats for data sharing. It also updates the context data for real-time use, and provides context query interfaces for context consumers.

(3) Context-aware service layer

With the context data obtained and stored in the context repository, the proactive context-aware computing service can be provided with the functional modules in this layer. For this project, the functional modules defined in this layer are:

a) *Context-aware service customisation* - The context-aware service that is customised to the contextual situation is provided with the rule-based method.
b) *Context-aware APIs* - The main context-aware service interfaces for context update, service request, context query, and user command are provided:
   contextUpdate.php?DATA=
   deviceControl.php?service=lightControl
   contextQuery.php?contextSet=
   userCommand.php?light=

c) *Mobile client applications* – The mobile client application provides user interfaces for manual control, permission and preference setting, and home facility monitoring, etc.

(4) Rule generation layer

The history context of the home environment is stored in the context repository for rule generation. The rough set theory-based method is implemented in this case study. New rules can be generated with mathematical models to revise the service customisation strategy.

With the hierarchical system architecture, the functions of the system are divided into separate modules explicitly. The modularisation of the system has split the work into reasonable parts, which simplifies the development and makes the system easy to maintain.

7.3.1.2 Wireless Connection and Data Communication

As discussed in 4.3, there are three alternative network topologies – infrastructure-based network, Ad Hoc network, and hybrid network. Concerning the data availability, simplicity, and stability of the network, the infrastructure-based network is selected for the connection of the devices. The structure of smart home network connection is as shown in Figure 8-14.

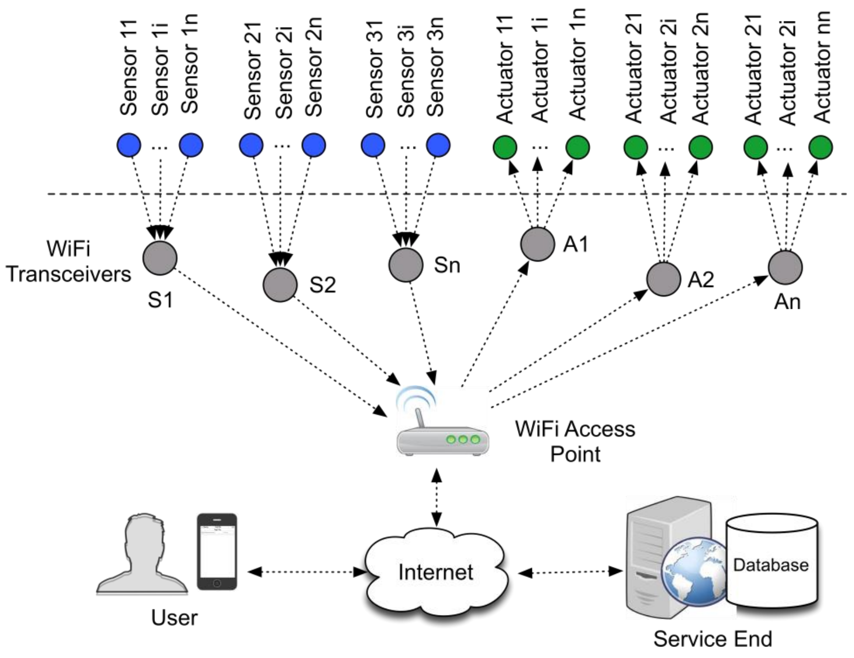


Figure 8-14 Network Connection of Smart Home System

In Figure 8-14, the wireless transceivers and mobile devices are connected with a WiFi access point for context update and context-aware service requests. The actuators are responsible for the execution of computing service for home facility control.

Since the context data is gathered and transmitted from the low-level electronics to the high-level wireless infrastructures, the interoperable data communication can be implemented with general and cross-platform standards and protocols. The main data communication procedures between devices are: sensor to MCU logic converter, MCU logic converter to WiFi transceiver, WiFi transceiver to remote server, and mobile to WiFi remote server.

- *Sensor to MCU logic converter* – This connection is achieved with low-level electronic control logic such as GPIO, I2C, 1-Wire Bus, etc.
- *MCU logic converter to WiFi transceiver*: The interface between MCU and WiFi adaptor in this system is RS-232 serial port.
- *WiFi transceiver to remote server*: This connection is through HTTP requests (POST, GET, PUT, DELETE), and example requests are as below.

  For context update:

  ```
  POST
  /contextUpdate.php?ID=01&brightness=v1&human=v2&obstacle=v3&humidity=v4&smo
  ke=v5&noise=v6&temperature=v7&location=v8
  ```

  For context-aware service request:

  ```
  GET /deviceControl.php?service=servicerequest
  ```

- *Mobile to remote server*: The HTTP requests (POST, GET, PUT, DELETE) are used as well, and example request is as below:

  ```
  NSMutableURLRequest *request = [[NSMutableURLRequest alloc] initWithURL:
          [NSURL URLWithString:@"http://host/folder/scriptfile.php"]];
  [request setHTTPMethod:@"POST"];
  NSString *postString = @"request=???";
  [request setValue:[NSString stringWithFormat:@"%d", [postString length]]
      forHTTPHeaderField:@"Content-length"];
  [request setHTTPBody:[postString dataUsingEncoding:NSUTF8StringEncoding]];
  [[NSURLConnection alloc] initWithRequest:request delegate:self];
  ```

The functions involving the data communication in the system such as context data acquisition and update, service request, user control with mobile device, and context-aware service execution are implemented with the above methods.

8.3.2 Context Acquisition and Context Modelling

With the system architecture, the network topology, and the methods for context data communication, the next step is to gather the context data and make it useful for the context-aware computing service.

(1) Sensors and context items

The embedded sensors employed to gather the environmental context in this case study are given in Figure 8-15. The sensors are connected to the embedded controllers and then interfaced to the wireless infrastructure through the WiFi transceiver.

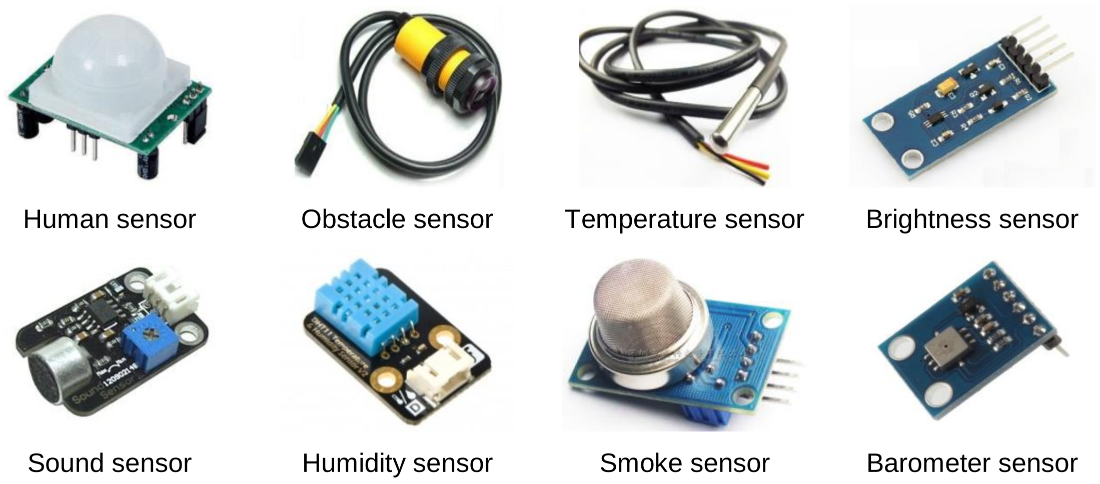


Figure 8-15 Embedded Sensors Employed in the System

The embedded sensors are used to gather different context values in the home area. Thus, they can be distinguished in many attributes such as resolution, power supply, signal output interface, etc. The embedded sensors and corresponding context items are given in Table 8-5.

Table 8-5 Embedded Sensors in the System

Sensor	Context Item	Denoted By
Brightness sensor	Brightness	C_brightness
Obstacle sensor	Obstacle	C_obstacle
Human sensor	Human	C_human
Humidity sensor	Humidity	C_humidity
Smoke sensor	Smoke	C_smoke
Sound sensor	Sound	C_sound
Barometer sensor	Barometric pressure	C_barometer
Temperature sensor	Temperature	C_temperature

The data acquisition with the embedded sensors is through the digital interfaces of the sensors such as GPIO, I2C, 1-Wire Bus, etc. The control of the acquisition is performed by the MCU. Generally, for the continuously varying sensors such as the temperature sensors, the context data is gathered and updated periodically in inquiry mode. For the sensors whose outputs are 1 and 0 such as human sensor, context is updated when output changes in interrupt mode.

In addition to the embedded sensors, there are also context items gathered in other manners such as mobile built-in sensor and software sensors. The available context items obtained with mobile built-in sensors and Internet providers are listed in Table 8-6.

Table 8-6 Context Obtained with Mobile Built-in Sensors and Soft Sensors

Sensor/Context Provider	Context Item	Denoted By
Microphone	User’s speech	C_speech
Camera	Object recognition	C_camera
GPS	Location	C_location
Accelerometer	User activity	C_acceler
3-axis gyro	Angular acceleration	C_gyroscope
Proximity sensor	Proximity to human body	C_proximity
Ambient light sensor	Brightness of ambient environment	C_amblight
Time	Current time	C_curtime
Weather provider	Weather information	C_weather

The context acquisition using the built-in sensors and soft sensors are implemented with the mobile devices. Some context data such as the current time and GPS location can be obtained directly, while some others such as accelerometer-based user activity recognition may include computation intensive tasks.

(2) Entities and context in the smart home system

The control of home facilities is determined by the predication of users’ intent, based on the context and the relations of entities in the home environment. Thus, to chart the relationship between the entities such as human, home facilities, and context in the system and design the rules for context-aware automation are essential tasks. The entities in the system are:

- *Human*: house member, housemate, neighbour, colleague, friend, and stranger
- *Houses*: neighbours, in the same town, in the same city, etc.

- *Home facilities*:

  Rooms of home: Living room, bedroom1, bedroom2, toilet, and kitchen

  Facilities in room: door, window, curtain, light, heater, fan, alarm, TV, etc.

- *Context items in use*:

  Environmental: brightness, humidity, smoke, temperature, sound, weather, etc.

  Time context: current time, time of day, and season

  User location: living room, bedroom, toilet, kitchen, and out of house

  User behaviour: watching TV, surfing the Internet, reading, sleeping, relaxing, cooking, etc.

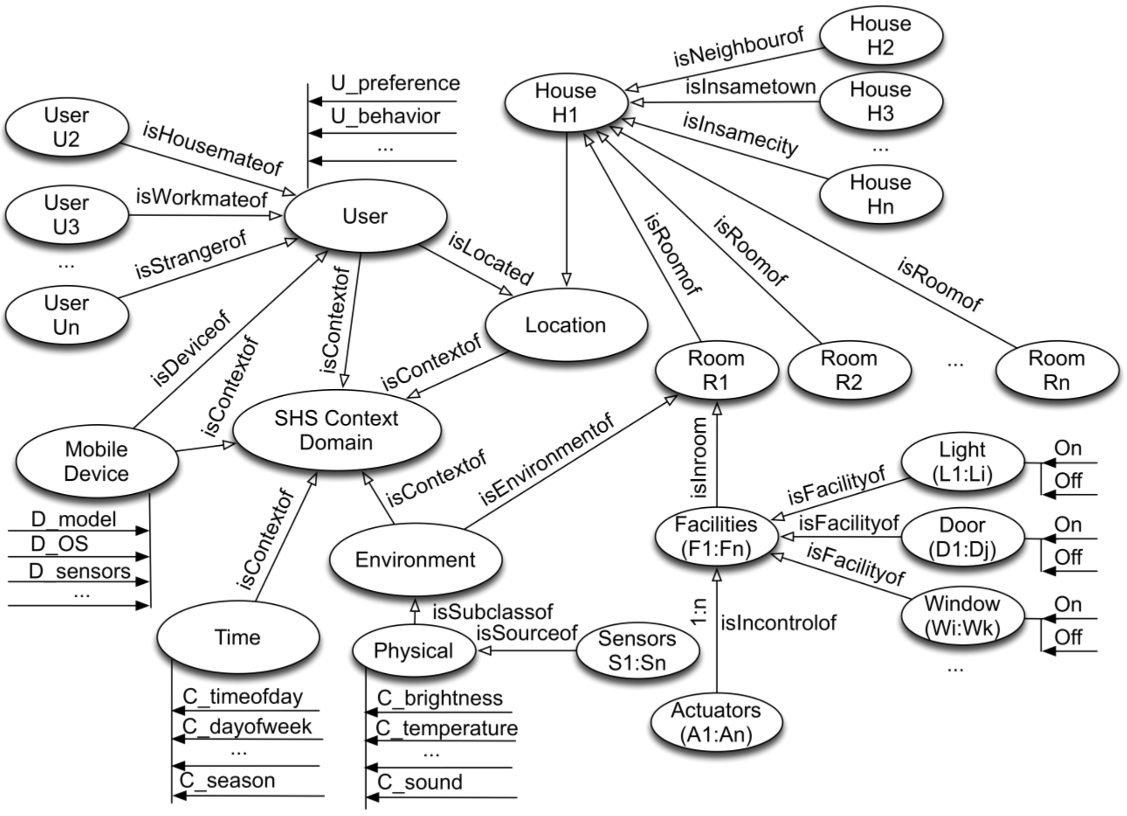


Figure 8-16 Model of Entities and Context in Smart Home System

The centre context terms presented in Chapter 5 is not comprehensive enough to express the application domain of the smart home case. It needs to be extended to detail the entities and their interrelationship. The model to describe the entities, context items, and their interrelationship for context-aware automation is given in Figure 8-16.

8.3.3 Results and Evaluation

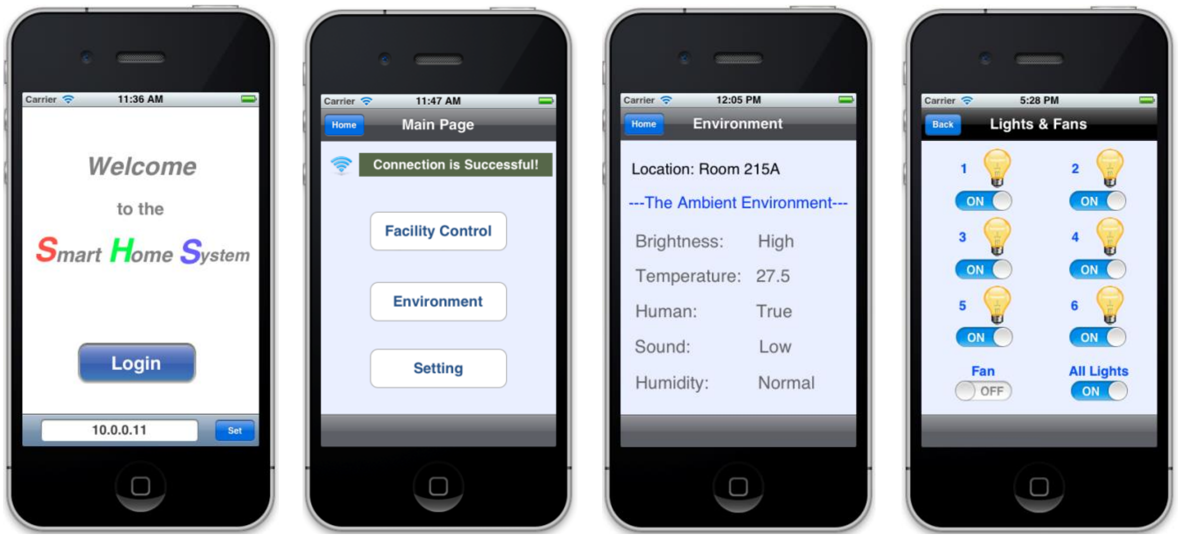
The system is built up successfully with the above mentioned techniques and context model. In order to demonstrate the feasibility of the proposed methods for context-aware computing service customisation, this section provides the results and evaluation of the system. The mobile client application for home facility control is introduced first, and then evaluation of the context-aware automation and rule generation methods is provided.

7.3.3.1 Mobile Client Application - Functionality and Interface

The mobile client application provides the users with an interface for home facilities state monitoring and manual control. It also provides an interface for users to set the attributes of the context data, such as permission of context to context consumers. In addition, the mobile application also gathers and updates context data during operation. The main functions of the mobile application are:

- Home facilities state monitoring and display
- Home facilities manual control
- Context data browsing
- Context item privacy setting
- Mobile device context sensing and updating

The function and interface of mobile client application is designed as discussed above, and it is implemented on the Apple iOS platform. It bridges the users and the computing system in the smart home environment, which collects and updates context data with mobile built-in sensors and performs users' control command using the HTTP POST/GET methods. Four selected sample interfaces of the prototype system on iOS devices are given in Figure 8-17.



10.0.0.11

Location: Room 215A

Brightness: High
Temperature: 27.5
Human: True
Sound: Low
Humidity: Normal

1 ON
2 ON
3 ON
4 ON
5 ON
6 ON
Fan OFF
All Lights ON

(a) Home page (b) Navigation panel (c) Context browsing (d) Display & control

Figure 8-17 Smart Home System Mobile Client Application Interface

In Figure 8-17, (a) is the login interface, (b) is the functional module navigation interface, (c) is the context browsing interface, and (d) is the facility state monitoring and manual control interface.

7.3.3.2 Semantic Distance-based Context-aware Service Customisation

Due to the complexity of the physical environment in the smart home system, the real-world evaluation of this system is very difficult. On the other hand, it involves a lot of techniques which also result in difficulties in experimental studies of the system. At the current stage, the prototype system is implemented with functions for rule-based control of commonly used home facilities. The table light and table fan control are selected as examples for the context-aware automatic control in this investigation.

The rules for home facility context-aware automatic control are listed in Table 8-7.

Table 8-7 Rules for Smart Home Context-aware Automation

ID	Rules (Conditions and Decisions Attributes)
Rule1	(C_{Human} is True)∧ ($C_{Brightness}$ is Ultra Low) ⇒ Light is On
Rule2	(C_{Human} is True)∧ ($C_{Brightness}$ is Low) ⇒ Light is On
Rule3	(C_{Human} is True)∧ ($C_{Temperature}$ is High) ⇒ Fan is On
Rule4	(C_{Human} is True)∧ ($C_{Temperature}$ is Ultra High) ⇒ Fan is On
Rule5	(C_{Human} is True)∧($C_{Temperature}$ is Ultra Low) ⇒ Heater is On
Rule6	(C_{Human} is True)∧ ($C_{Temperature}$ is High) ∧ ($C_{Weather}$ is Sunny) ⇒ Window is Open

In Table 8-7, take *Rule1* for example, if human is detected by sensors (C_{Human} *is True*) and the ambient brightness is ultra low ($C_{Brightness}$ *is Ultra Low*), the light is turned on automatically. Provided the condition attributes in the smart home system are:

$$A=\{C_{Human}, C_{Brightness}, C_{Temperature}, C_{Noise}, C_{Weather}, C_{Humidity}, C_{Smoke}, C_{Time}\}$$

The decision attributes for home facility control:

$$D=\{d_{Light}, d_{Fan}, d_{Heater}, d_{Window}\}$$

is decided according to the semantic distance between the context sample and the rules.

In order to demonstrate how decision making works in smart home system, a set of context data in database of the prototype system is selected as the current context:

$$CV_i=\{True, Low, Low, Low, Rainy, Low, Low, Morning\}$$

The *dist(CV_i, Rulei)* denotes the semantic distance between the context sample CV_i and rule *Rulei*. According to Figure 8-16, the temperature in the tree hierarchy indexing structure in the context model can be represented with the diagrams in Figure 8-18.

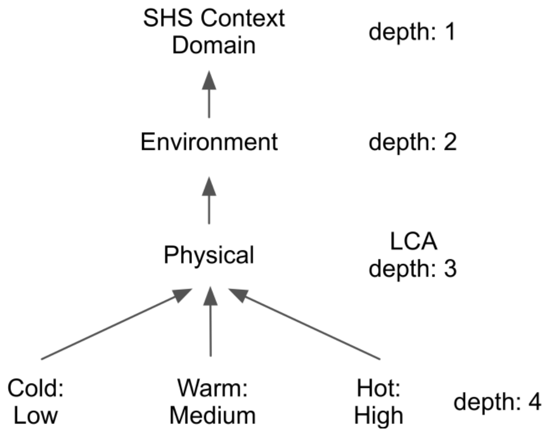


Figure 8-18 Tree Hierarchy Structure of Temperature in SHS Context Domain

Then, according to rule matching presented in section 6.3, the normalised semantic distance between the context sample CV_i and the rules in Table 8-7 can be calculated.

For example, in the sample data $CV_i[C_{Temperature}]$ = *Low*. According the tree hierarchy structure, *depth(LCA(Cold:Low, Hot:High))* = 3, *depth(Cold:Low)* = 4, *depth(Hot:High)* = 4. According to formula (6-8) and (6-9),

$$\text{dist}(CV_i[C_{Temperature}], Rule3[C_{Temperature}]) = 1 – 2*3/(4+4) = 0.25$$

With the same method:

$$\text{dist}(CV_i[C_{Human}], Rule3[C_{Human}]) = 1 – 2*4/(4+4) = 0$$

Then, accordint to (6-10),

$$\text{dist}(CV_i, Rule3) = 0.5*\text{dist}(CV_i[C_{Temperature}], Rule3[C_{Temperature}]) + 0.5*\text{dist}(CV_i[C_{Human}], Rule3[C_{Human}]) = 0.125$$

where 0.5 is the weight of the two relevant context items.

So, the results of the rule matching are given as follows:

(1) $dist(CV_i,Rule1) = (1-2*4/(4+4) + 1-2*3/(4+4))/2=0.125$

(2) $dist(CV_i,Rule2) = (1-2*4/(4+4) + 1-2*4/(4+4))/2=0$

(3) $dist(CV_i,Rule3) = (1-2*4/(4+4) + 1-2*3/(4+4))/2=0.125$

(4) $dist(CV_i,Rule4) = (1-2*4/(4+4) + 1-2*3/(4+4))/2=0.125$

(5) $dist(CV_i,Rule5) = (1\text{-}2*4/(4+4) + 1\text{-}2*3/(4+4))/2=0.125$

(6) $dist(CV_i,Rule6) = (1\text{-}2*4/(4+4) + 1\text{-}2*3/(4+4) + 1\text{-}2*3/(4+4))/3=0.167$

In this system, the weight of the attributes in the rules are equal, the weight is *1/n* when there are *n* attributes in a rule. The semantic distance between *CVi* and the rules are presented in Figure 8-19.

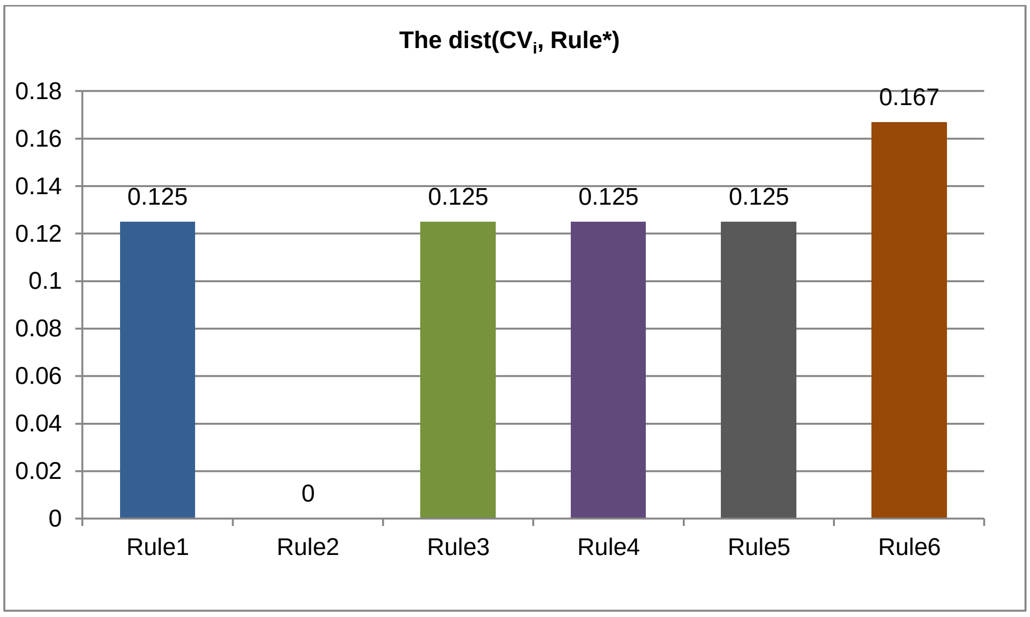


Figure 8-19 Semantic Distance between CV_i and the Rules in SHS

It is clear that the minimum semantic distance between CV_i and the pre-defined rules is $dist(CV_i,Rule2)$. Thus, *Rule2* is considered to be the appropriate rule for the home facility control, and the relevant service 'Light is ON' is executed automatically.

In order to evaluate the efficiency of the rule matching algorithm as a service application with the above use case, it is tested on two testing platforms: (1) Local host Apache/MySQL/PHP (XAMPP 1.82) server on an iMac machine with 2.66GHz Intel Core 2 Duo CPU and 4G 1067MHz DDR3 RAM; and (2) Remote Apache/MySQL/PHP (XAMPP 1.82) server as well shipping Ubuntu 12.04 with Intel Core i7-3770 CPU@3.40GHz×8 and 16GB 1066 MHz DDR3 RAM. The testing results on the two platforms are as shown in Figure 8-20.

From the figure, time consumption of the rule generation algorithm is nearly in line with the number of rules on both the two testing platforms. But the performance of the algorithm is better on the remote testing server than on the iMac local host server, because of the difference in computation power. It needs only 0.0267 seconds for decision making of the computing service on the remote testing server with 100 rules. That is to say the real-time performance of the rule matching process can be guaranteed in practical use.

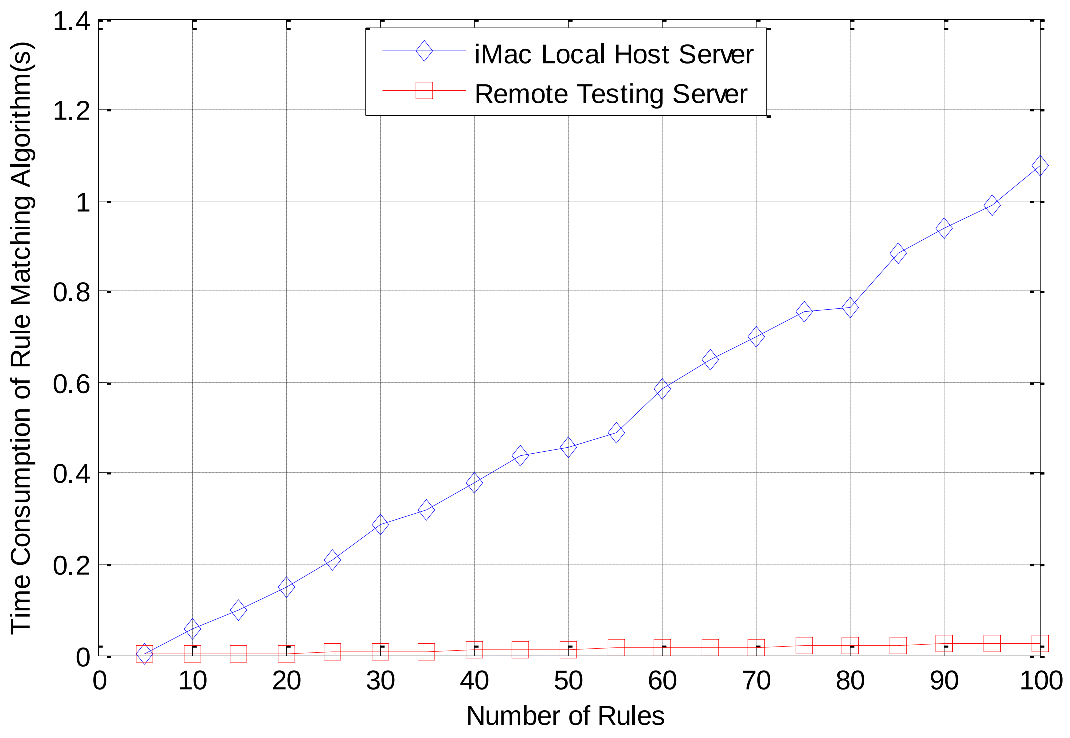


Figure 8-20 Performance of Rule Matching Algorithm

With this method, the appropriate context-aware computing service can be obtained by calculating the semantic distance between the rules and the observation of current context.

7.3.3.3 Rough Set Theory-based Rule Generation

In the smart home system, the context-aware automatic control can be achieved by rule-based adaptation with the explicitly defined rules. Because the context changes dynamically, it is difficult to define the complicated relationship between the entities, computing service, and the multifaceted context comprehensively. Therefore, the rule-based system may not be able to satisfy all the use scenarios due to the variation and diversity of the entities and context. In order to endow the system with the learning capability to be adapted to the entities and context change, the rule generation method is expected for appropriate computing service adaptation. In order to demonstrate the feasibility of the RST-based rule generation method presented in section 6.3.2, an experiment is conducted in this section.

Because of the uncertainty attribute of users' preference, it is quite difficult to chart the rules from the complex interrelation between the users, facilities, and context items. In this prototype system, the rule generation method is demonstrated through the table light control use case with specific context items listed in Table 8-8.

Table 8-8 Context Items for Rule Generation

C1: Brightness(B)	L – Low M – Medium H – High	C5: Noise(N)	L – Low H – High
C2: Human(HU)	T – True F – False	C6: Humidity(HD)	L – Low H – High
C3: Weather(W)	S – Sunny R – Rainy C – Cloudy	C7: Smoke(S)	L – Low H – High
C4: Temperature(TP)	L – Low M – Medium H – High	C8: Time(TM)	M – Morning A – Afternoon E – Evening

The user operations and the corresponding context data in the context repository of smart home prototype system selected for rule generation are listed in Table 8-9, in which LON denotes 'Light ON' and LOFF denotes 'Light OFF'.

Table 8-9 Records of History Context and User Operations

ID	**Context Items and Values**								**User Operations/ Approved service**
	C1 (B)	C2 (HU)	C3 (W)	C4 (TP)	C5 (N)	C6 (HD)	C7 (S)	C8 (TM)	
1	L	T	S	H	L	L	L	E	LON
2	H	T	S	H	H	L	L	M	LOFF
3	L	T	C	H	L	L	L	A	LON
4	H	T	S	H	L	L	L	A	LOFF
5	L	T	C	H	H	L	L	M	LON
6	L	F	C	H	H	L	L	M	LOFF
7	L	T	C	H	L	L	L	A	LON
8	H	T	S	H	L	L	L	A	LOFF
9	L	T	C	H	L	L	L	M	LON
10	H	T	S	H	L	L	L	M	LOFF
11	L	T	R	M	L	H	L	M	LON
12	H	F	S	H	H	L	L	A	LOFF
13	L	T	R	H	L	H	L	A	LON
14	H	T	S	M	L	L	L	M	LOFF
15	L	T	S	M	L	L	L	E	LON
16	H	T	S	H	L	L	L	A	LOFF
17	L	T	R	H	L	H	L	M	LON
18	H	T	S	M	L	L	L	M	LOFF
19	L	T	S	H	H	L	L	M	LON
20	H	T	S	H	L	L	L	M	LOFF
…				…					…

According to formula (6-13), for the first row in Table 8-7, *DM(1,1)=0*, since decision attribute *d(1)=d(1).* For the second row, $DM(1,2)=[C_1,C_5,C_8]$ represented as $C_1C_5C_8$, which means the values on the 1st, 5th, and 8th attributes are different when decision $d(1)\neq d(2)$.

With the same method, the discernibility matrix $DM_{20\times 20}$ for the 20 context value samples in Table 8-9 can be obtained:

$$\begin{array}{c|cccccccccccc|}
 & 1 & 2 & 3 & 4 & 5 & 6 & \dots & 16 & 17 & 18 & 19 & 20 \\
\hline
1 & 0 & & & & & & & & & & & \\
2 & c_1c_5c_8 & 0 & & & & & & & & & & \\
3 & 0 & c_1c_3c_5c_8 & 0 & & & & & & & & & \\
4 & c_1c_8 & 0 & c_1c_3 & 0 & & & & & & & & \\
5 & 0 & c_1c_3 & 0 & c_1c_3c_5c_8 & 0 & & & & & & & \\
6 & c_2c_3c_5c_8 & 0 & c_2c_5 & 0 & c_2 & 0 & & & & & & \\
\vdots & & & \vdots & & & & \dots & & & & & \\
16 & c_1c_8 & 0 & c_1c_3 & 0 & c_1c_3c_5c_8 & 0 & & 0 & & & & \\
17 & 0 & c_1c_3c_5c_6 & 0 & c_1c_3c_8c_6 & 0 & c_2c_3c_5c_6 & & c_1c_3c_8c_6 & 0 & & & \\
18 & c_1c_4c_8 & 0 & c_1c_3c_4c_8 & 0 & c_1c_3c_4c_5 & 0 & \dots & 0 & c_1c_3c_4c_6 & 0 & & \\
19 & 0 & c_1 & 0 & c_1c_5c_8 & 0 & c_2c_3 & & c_1c_5c_8 & 0 & c_1c_4c_5 & 0 & \\
20 & c_1c_8 & 0 & c_1c_3c_8 & 0 & c_1c_3c_5 & 0 & & 0 & c_1c_3 & 0 & c_1c_5 & 0
\end{array}$$

According to algorithm 2, when the discernibility matrix is simplified to **0**, the reduction *Red={c1,c2}* is obtained. Therefore, the rules can be generated accordingly:

(1) $(c_1 = Low) \wedge (c_2 = True) \Rightarrow$ *Light is On*

(2) $(c_1 = High) \wedge (c_2 = True) \Rightarrow$ *Light is Off*

(3) $(c_1 = Low) \wedge (c_2 = False) \Rightarrow$ *Light is Off*

(4) $(c_1 = High) \wedge (c_2 = False) \Rightarrow$ *Light is Off*

Taking Rule (1) as an example, it represents that if the ambient brightness is low and user is in the home area, the light will be turned on. The rules created in the rule generation can then be added to the rule library to supervise the context-aware automation service. Therefore, the system is endowed with the ability to generate the context-aware automation rules and the capability to adapt the system to the dynamic context environment.

In order to evaluate the efficiency of the rule generation algorithm, it is tested as a service application with the use case of light control. The algorithm is tested on two testing platforms, iMac local host server and remote testing server, as mentioned above. The testing results are as shown in Figure 8-21.

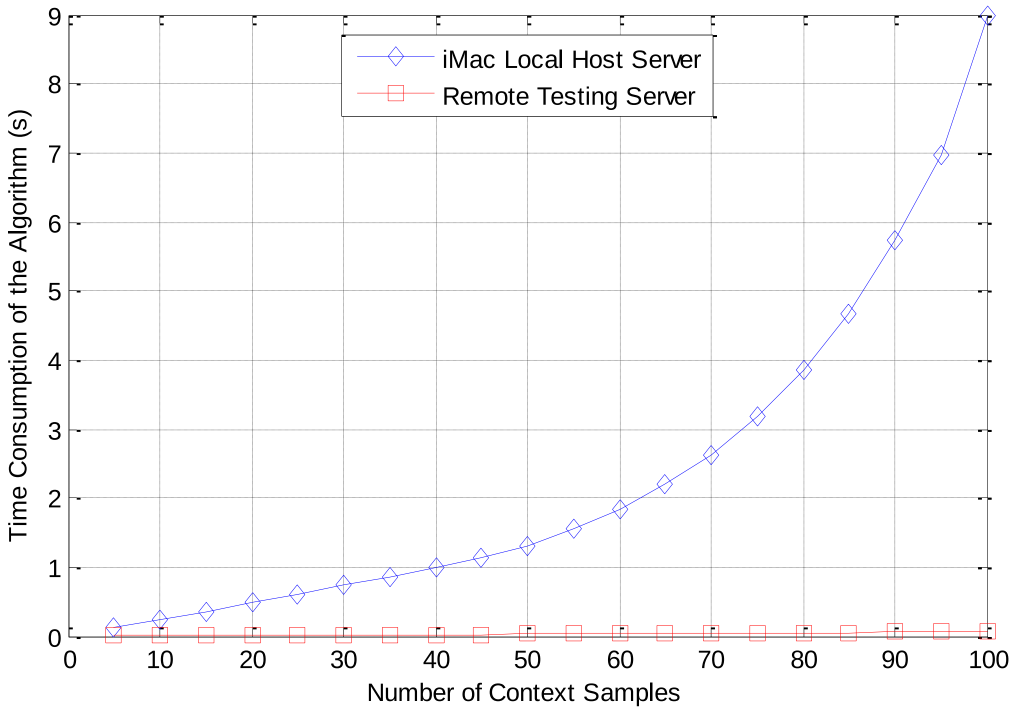


Figure 8-21 Performance of the Rule Generation Algorithm

Figure 8-21 illustrates the relationship between time consumption and number of context samples of rule generation algorithm with 8 context attributes for the above use case. It is not difficult to find that the performance of the rule generation algorithm behaves differently on the two platforms. The time consumption is too long (8.99 seconds) to be used for computing service adaptation for iMac local host server, while it is acceptable (0.067 seconds) for the high performance remote testing server when the quantity of context sample is 100. That is to say, performance of the algorithm is highly related to the computation power of testing platforms.

In order to gain an in-depth view of the rule generation algorithm on the two platforms, the rule generation process is divided into two procedures: (1) data initiation, and (2) rule generation algorithm. The data initiation procedure is the process of database data query and memory initiation, and rule generation algorithm is the process of rule generation with the prepared data. The two procedures of the rule generation algorithm for the testing platforms are given in Figure 8-22 and Figure 8-23.

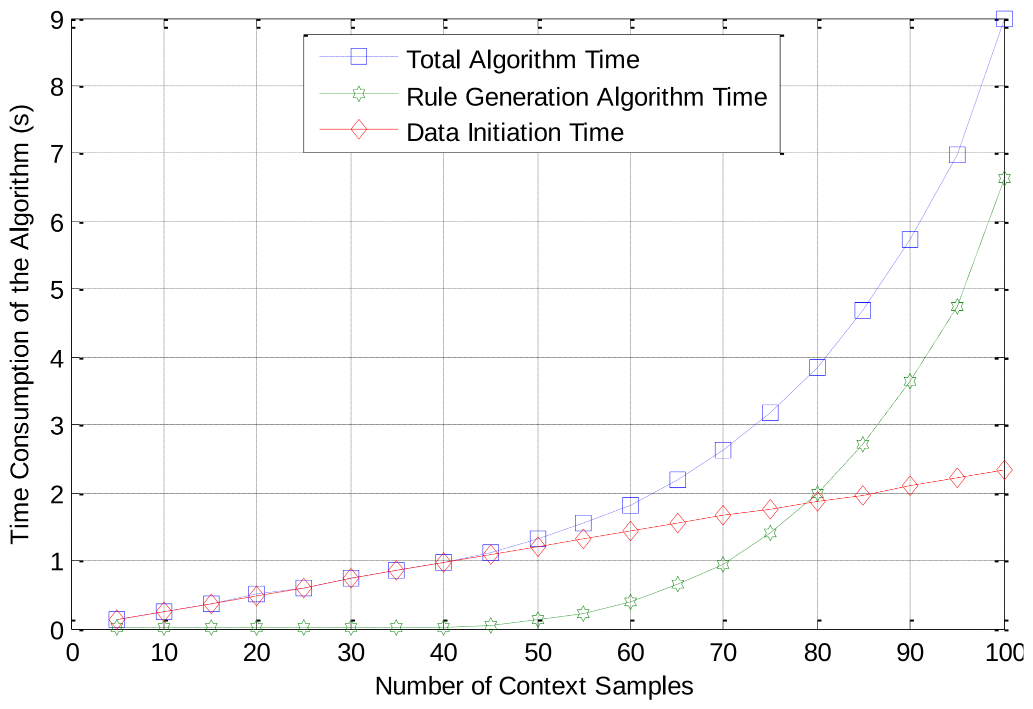


Figure 8-22 Performance of the Rule Generation Algorithm on iMac Local Host Server

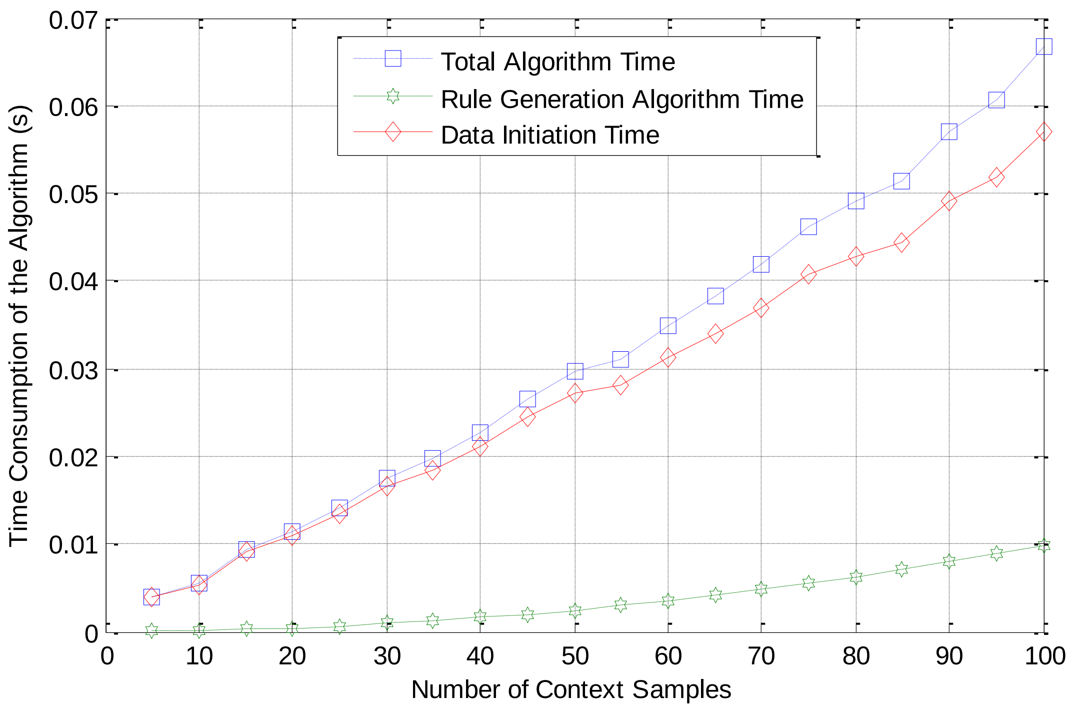


Figure 8-23 Performance of the Rule Generation Algorithm on Remote Testing Server

From Figure 8-22, the rule generation is fast when the number of context samples is not large especially when the number of context samples is less than 40, while the data initiation time is much longer. But with the increase of context samples over 50, time consumption of the rule generation algorithm increases quickly. When the number of context samples is over 80, the rule generation time is longer than data initiation time and the difference increases steeply with the number of context samples. In the whole process, the data initiation time increases with the number of context samples linearly, while the rule generation time has a nearly exponential relationship with the number of context samples.

From Figure 8-23, both data initiation time and rule generation algorithm time on the remote testing server are very short and they are nearly in line with increase of context sample number. The rule generation algorithm time is shorter that the data initiation time to a large extent. With 100 context samples, the total time consumption is only about 0.067 second. Therefore, the real-time performance can be therefore guaranteed for practical use.

Obviously, the varying trend of the curves is different on the two platforms. That is because of their difference in computation power of the test platforms. When the computation power is strong, the time consumption is in line with the quantity of context samples. However, the time consumption is in exponential growth when the computation power is not strong enough.

From the above discussion, we can say that the rule generation algorithm is fast enough for practical use provided the service platform is powerful enough. The computation power of the platform, efficiency of the server database data access, and number of context samples are key factors influencing the speed of rule generation process.

8.3.4 Summary

Normally, the computation tasks are closely related to users' preference and habit which are not easy to be obtained directly, and the practical home environment is usually very complex. This case study conducts a proof of concept demonstration of the proposed context-aware computing service customisation strategy including both the computing service decision making and the rule generation methods.

With experiments on the commonly used home facility control, the context-aware service customisation and rule generation methods are implemented and good results are produced. In the smart home environment, the interrelationship between the entities is presented with the context model for context-aware service provisioning and the structure may be very complex. Meanwhile, the data structure, accuracy, source, and validation time of context may differ significantly. The heterogeneous data requires the service customisation strategy to be able to effectively handle the complicated data and provide the appropriate computing service in real-time. The semantic distance-based rule matching method is strong in handling context data of different data types and structures, both numeric and tree-hierarchy indexing terms in ontology describing complicated relationships. Therefore, it is appropriate for the rule matching of context data, and it is easy to implement in the service end of computing systems.

For the rule generation method, the rough set theory is a mathematic method strong in handling information with some degree of uncertainty. In this case study, it successfully determines the key context attributes affecting the result of the context-aware service decision making. From the experimental study, results show that the rule generation algorithm is fast enough for practical use provided the service platform is competent in computation power. The computation power of the platform, efficiency of the database data access, and number of context samples are key factors influencing the speed of rule generation process. Theoretically, the more context value samples involved in the rule generation, the more complete and accurate the rules for the use case will be. But more computation load and risk of failures in real-time processing may be brought to the system. Therefore, how the proper quantity of context samples is selected to create effective rules efficiently and then integrate the generated rules in the rule library is important for context-aware systems.

8.4 Discussion

8.4.1 Achievements and Strengths

The case studies presented in this chapter have both integrated the context information as implicit input for service customisation with the proposed hierarchical system architecture and service customisation strategy. Case study 1 employs the mobile device built-in sensing techniques to improve the usability and operational efficiency of mobile applications, and case 2 utilises the commercial available low cost sensing techniques to assist people’s daily life by home facilities context-aware automation.

Through case study 1, it is found that the hierarchical context-aware architecture with the mobile device built-in sensing technique is a promising way to assist the user interaction of mobile applications. The context-aware mobile applications can effectively simplify user operations and enhance the efficiency of operations. One underlying problem is the heterogeneity in the mobile device software platforms. For example, although the four platforms all support camera-based barcode scanning, there is no software development kit that supports all platforms and they need to develop for each of them individually. Another potential problem is the divergence in hardware of the mobile devices. The quality of mobile devices and hardware modules significantly influences the performance of the context-aware computing applications. The heterogeneous hardware and software platforms require a standard strategy to exploit the mobile built-in sensors flexibly and efficiently.

Through case study 2, the context acquisition, context handling, context-aware service customisation, and rule generation methods are implemented. The ambient context information is gathered with the embedded sensor technologies and saved into the database for the automation of home facilities. The decision making method provides the appropriate command with the semantic distance based rule matching efficiently, and rule generation method determines the key context items and generates the rules successfully. The results and the performance evaluation show that it is feasible to employ the rule-based service customisation and RST-based rule generation method for computing service customisation.

The hierarchical system architecture designed in this investigation divides the techniques of different subject domains into different layers with interoperable interfaces between the layers. The different layers also deal with context information of different sematic levels. Compared to the existing architectures discussed in section 2.4.2, this loose coupling architecture is flexible to expand and easy to maintain. It is also strength for this architecture to deal with the context information of different semantic level since they can be processed with similar methods in individual layers. The highlight which distinguishes the proposed architecture with the others is its rule generation layer taking advantage of history context to generate rules to supervise the service customisation.

The proposed computing service customisation strategy separates the two steps of computing service decision making and rule generation. The rule matching method can be used for decision making of computing service using the current context, and the rule generation method is then used to determine new rules for service customisation. Since the computation in this process is only to calculate the semantic distance between the context set and a limited number of rules in the rule set, the computation burden of the decision making is lightweight. Compared with some existing methods, this method can get rid of the limitation of traditional IF-THEN logic and make the system capable of determining the rules with history context. In addition, the semantic distance based method explicitly defines decision making with Manhattan distance and GCSM, which is suitable for the diverse data types and structures of context compared with FMADM, Multi-facet Item based Method, and MAUT. It also needs fewer context samples and less computation compared with the probability-based method, which requires numerous context samples to guarantee accuracy. Then, compared with the ANFIS and Rough-Fuzzy method, the separation of service decision making step and rule generation step of the proposed strategy makes it lightweight in computation and easy to implement.

8.4.2 Weaknesses and Open Issues

Although the proposed methods are implemented with case studies, this investigation can be said in a proof of concept demonstration stage, since only specific computation tasks are implemented to prove the feasibility of the methods employed. The lack of comprehensive evaluation of the effectiveness of the context-aware service and accuracy of the rules generated is a typical weakness of this work.

In addition, there are other open issues worth further investigation:

(1) Context-aware applications are normally domain-specific, and the performance of context-aware systems computing is largely dependent on its understanding of context data describing the real world entities. Therefore, the method to bridge the gap between the real world entities and context data in context models with domain knowledge for context sharing and understanding is significant.

(2) Practically, context items may make different contributions in computing service decision making, it is necessary to determine the weight of the context items and distinguish them by the weight.

(3) The method of selecting the proper quantity of context samples for rule generation, considering effectiveness of generated rules and real-time performance of the system, is an important task for context-aware systems.

Apparently, it is a significant research area of context-aware computing to involve the context data as implicit input to help the computer systems to understand human requests and then provide the appropriate computing service. The progress in this area is mostly prompted by the emergence of novel techniques and technical advances in related fields. In the future, potential problems such as the above mentioned issues may gain more research attention.

8.5 Summary

In this chapter, the proposed system architecture and methods are put into practice with case studies. Case study 1 demonstrates the feasibility of employing the sensing techniques to improve the usability and efficiency of mobile applications. The user operation is simplified and operational efficiency is effectively enhanced. Case study 2 then offers proof of concept demonstration of the context-aware service customisation and rule generation methods. The appropriate computing service can be determined with the rule matching method and the service customisation rules can be obtained efficiently with the rule generation method. Both case studies show great potential in adapting the computing service to the contextual situations with sensing techniques, but they also reveal some open issues for further investigation.

Chapter 9 Conclusion and Future Work

In brief, this thesis presents design-based research to demonstrate the proposed context-aware computing architecture and technical approaches integrating context information into computation to customise the computing service to the contextual situation. Instead of solely proving the feasibility of the novel sensing techniques, this investigation focuses on framework support that bridges the gap between sensor data and applicable context, and the models and algorithms to customise computing service with relevant context. Despite the complexity of the experiments being somehow constrained by the sophisticated situation, heterogeneous techniques, human involvement, objective and convincing results are obtained in the case studies. In this final chapter, the contributions of this investigation are justified, some open issues are highlighted and future work is suggested accordingly.

9.1 Conclusion

In this thesis, Chapters 4, 5, and 6 present the system architecture, techniques, and methods of context-aware mobile computing, which are implemented in case studies in Chapter 8. It can be concluded that the hypothesis of employing the mobile built-in and embedded sensing techniques to customise the computing service to the contextual situation is feasible and it is a promising research area. The hierarchical system architecture, context modelling and acquisition methods, context-aware service customisation strategy, and rule generation mechanism presented in this thesis are appropriate for the context-aware computing systems based on the mobile built-in and embedded sensing techniques.

In this section, we conclude this thesis by summarising the contributions of the investigation. The main contributions of this thesis fall on the following aspects:

(1) *A General Hierarchical Context-aware System Architecture for Interoperable Use of Context with a Rule Generation Layer for Further Use of Context*

Rather than relying on some domain specific framework or middleware techniques, this investigation proposes a hierarchical context-aware system architecture with explicitly defined functional modules as a generalised design concept for context data handling and service provisioning. Due to the heterogeneity of techniques in the physical layer, efficiency and interoperability become critical problems. Therefore, the openness and loose coupling characteristics of this hierarchical architecture can minimise the complexity and cost in application development. In addition, this architecture highlights the further use of context by employing history context to generate rules to supervise the computing service customisation. The system architecture and design principles are implemented in case studies in Chapter 8 and the expected results are achieved in the development.

(2) *An In-depth Analysis and Classification of Context for Mobile Built-in and Embedded Sensors and the Corresponding Acquisition Methods*

In order to seek the appropriate methods for context modelling and context acquisition, the context of mobile device-based context-aware applications is analysed and classified, considering attributes of context and the potential use of the latest emerging sensing techniques. According to the classification, the context acquisition methods are discussed with examples in Chapter 5 to obtain the various distributed information in the dynamic environment and use it as context. This investigation classifies context acquisition methods into embedded sensor acquisition, mobile built-in sensor acquisition, Internet retrieval, and user manual input. This classification has broadened the scope of context, which used to be limited to mobile built-in sensors.

(3) *Context Modelling and Context Representation Methods Suitable for Mobile Device-based Use Cases*

An open and extensible context model with the centre context terms for prototyping of complicated context-aware applications is presented in section 5.2.2, according to the classification and analysis of context. On the other hand, the lightweight and interoperable context data representation format presented in section 5.3.2 can enhance the understanding of context in the computing system, which actually promotes the efficiency of context distribution and context processing in context-aware systems. Both methods are appropriate for resource limited mobile computing devices. The context modelling and representation methods presented in this investigation are implemented in the case studies in Chapter 8.

(4) *A Rule-based Context-aware Service Customisation Strategy for Context-aware Computing Service Provisioning*

The proposed functional model for context-aware service provisioning is considered as a decision information system, where context variation and user operation are inputs and the corresponding computing service is output. The context-aware adaptation engine compares the current context with the rule set using semantic distance-based rule matching to seek the particular rule for decision making. This algorithm is competent for both numeric and complex data structure, which is suitable for context-aware applications in complex contextual situations. Implementation in Chapter 8 has demonstrated the feasibility and performance of the context-aware customisation method.

(5) *A Rule Generation Mechanism Taking Advantage of History Context to Supervise the Service Customisation*

A rule generation mechanism based on the rough set theory is used to fulfil the rule generation with the history data consisting of context and approved service in context repository. The employed rough set theory-based rule generation method is suitable to deal with the imprecise, uncertain, and incomplete context information. Therefore, this context-aware system is allowed to determine the potential rules with rule generation method, which makes the system open and flexible to learn the situation change. Case studies 1 and 2 in Chapter 8 demonstrate the feasibility and performance of the attribute reduction and rule generation methods with experimental study.

The above mentioned architecture, methods, models and algorithms are implemented in case studies in Chapter 8. Case 1 employs the novel mobile built-in sensing techniques to enhance the efficiency of user interaction in mobile applications. It focuses on the context data acquisition and service customisation to reduce tedious manual operations, which has largely improved the real-time performance of the mobile application. Case 2 employs mobile built-in and embedded sensing techniques to facilitate home appliance automatic control. It focuses on service customisation and rule generation using the history data consisting of context and user operations, which provides a support mechanism for the discovery of potential rules to better predict users' expectation and tailor the computing service. Apparently, both the applications in the case studies are enhanced with the presented methods.

9.2 Future Work

In design and implementation of context-aware systems, it also reveals some open issues which deserve further investigation. On one hand, the context-aware mobile applications usually involve a wide range of techniques and methods in several research fields, which inevitably raise challenges in interoperability and efficiency of context handling. On the other hand, the context involved in the experimental studies may not be sufficiently comprehensive and accurate to characterise the user and physical environment which affect the computing service. The main aspects recognised in this research are listed as follows:

(1) *Comprehensive and Accurate Description of Contextual Situation*

Uncertainty and inaccuracy are typical characteristics of various context data. Thus, methods of improving the stability of the sensor acquisition, such as failure detection and tolerances strategies, need to be integrated into the context handling framework. This investigation integrates the meta-information to describe the QoS, which is still at a stage of proof of concept and requires in-depth study. The relevant context needs to be thoroughly provided in order to effectively characterise the user and physical environment for context-aware service customisation towards appropriate computing service.

(2) Extension of Centre Ontology and Ontology-based Computation

The ontology-based context model with a centre ontology presented in Chapter 5 is application domain independent. However, the context-aware mobile systems are application specific, for which the computing service is decided by particular context items. Thus, it is necessary to define the entities and context items explicitly to obtain the appropriate computing service. On the other hand, for the cross-platform sharing of context data, the semantic description of device profile, shared data, and tasks are expected to be easily understood by the computing systems. Therefore, the ontology-based context presentation of real world entities and ontology-based computing for context-aware computing service is a prospective research area.

(3) Security and Privacy Strategies to Protect Users' Sensitive Context Information

The context data may sometimes contain users' sensitive information, such as confidential data, basic personal information, contact details, and preference. On the other hand, the context needs to be distributed and shared in the context-aware systems. Therefore, relevant strategies are expected to guarantee security and privacy when pursuing the performance of context-aware applications in an interoperable framework.

(4) The Insight into the Rule Generation Mechanism

For the DIS consisting of context and approved service, attribute reduction and rule generation based on the discernibility matrix-based rough set theory is implemented. The rule update method for the revision of existing rules is an important task to achieve in the future work, and the accuracy of rule generation and update needs to be evaluated with a rational scheme. Besides, it is significant to investigate the method to select the proper quantity of context samples for automatic rule generation and update, considering accuracy of rules and real-time performance of the whole system.

(5) Full Evaluation Strategy for the Context-aware Systems

The evaluation of context-aware applications is usually particularly difficult due to the constraints resulting from some objective factors. The case study systems in this investigation also lack exhaustive evaluation. The systems will be fully evaluated to obtain objective data and more persuasive results. For instance, case study 1 will be evaluated with more users of different ages, occupations, etc., and case study 2 will be evaluated with sufficient time to determine how the context-aware service satisfies users' expectations.

The above aspects are the critical issues suggested for further investigation, and they are the main constraints keeping the context-aware system from widespread use. Although various new techniques are continuously emerging, the challenges facing context-aware computing systems are still the design principles, fundamental methods and theories. To put it briefly, the focal objectives of the future works are: (1) to accurately characterise the users and physical environment with the various sensing techniques; (2) to enhance the sharing and understanding of context with domain knowledge; (3) to model and express the contextual situation in a computable model in the computer systems; and (4) to determine the underlying knowledge more efficiently to effectively supervise the service provisioning.

9.3 Summary

In summary, this investigation identifies the problems of lacking generalised architecture and context-aware service customisation strategy, and then employs relevant solutions with demonstration using the design research method. The essential contribution is the hierarchical context-aware system architecture as generalised design concepts, and the context-aware service customisation strategy with rule generation mechanism for high-level supervision. Despite the difficulties of the context-aware system in practical use, the implementation and evaluation of the case studies are successful as proof of concept demonstrations. The presented methods can be proved to be feasible and practical for context-aware application development.

In the predictable future, the technical advances in IoT, WSN, and cloud computing will promote the various computing terminals to be interconnected in a pervasive paradigm to fulfil service automation or intelligent decision support. The design concept and methods for context-aware system development presented in this investigation can offer some reference for the envisioned future computing systems.

ACKNOWLEDGEMENTS

This thesis would not have been possible without the guidance and support from a number of people, who have contributed to the success of this research work directly or indirectly. My sincere appreciation goes to all of those who have offered their kind help during my PhD study.

My first and foremost sincere appreciation goes to my supervisor Professor Joan Lu for her guidance, encouragement, and continuous support throughout my PhD study. Her support makes it possible to pursue this research and complete the thesis successfully. Her strong motivation and persistent attitude to research have inspired me all the time during my study.

In the second place, I would like to express my gratitude and respect to Dr. Qiang Xu for his patience, insightful comments, and encouragement in the supervision of my research work.

In addition, I am grateful to all the members of the XDIR research group: Dr. Raoul Pascal Pein, Dr. Lan Wang, Dr. Zhiyong Zhang, Mr. Wei Guo, Mrs. Jinyan Li, Mr. Omar Ghabar, and Mr. Jiayi Zhang for their open-mindedness and creativity in making such a pleasant research atmosphere. Everyone has positively influenced the research work through discussion, encouragement, and constant support at difficult times.

In addition to academic assistance, I would like to thank my parents, my sister, and my girlfriend for their immeasurable sacrifices and selfless love. It is their consistent support and belief in me throughout the years of my study that has enabled me to go as far as I have. My deep appreciation goes to my girlfriend Miss Ying Song for her kind comprehension during the writing up of my thesis. Her sincere trust gives me confidence and endless strength, and her attentiveness and consideration make me feel easy and pleasant at such a hard time.

Finally, I appreciate the support from the European Commission and the University of Huddersfield, who funded part of the research work presented in this thesis.

LIST OF PUBLICATIONS

[1] Meng, Z., and Lu, J. 2014. A Rule-based Service Customisation Strategy for Smart Home Context-aware Automation. *Under Review. Submitted to IEEE Transactions on Mobile Computing.*

[2] Meng, Z., and Lu, J. 2013. Integrating Technical Advances in Mobile Devices to Enhance the Information Retrieval in Mobile Learning. *International Journal of Information Retrieval Research*, 3(3). pp. 1-25

[3] Meng, Z. and Lu, J. 2013. Integrating Smartphone's Intelligent Techniques on Authentication in Mobile Exam Login Process. In: *5th International Conference on Computational Collective Intelligence Technologies and Applications,* 11/13 September 2013, Craiova, Romania. pp. 130-142

[4] Meng, Z. and Lu, J. 2011. Implement the Emerging Mobile Technologies in Facilitating Mobile Exam System. In: *2nd International Conference on Networking and Information Technology*, 25/27 November 2011, Hong Kong, China. pp. 80-88

[5] Meng, Z. and Lu, J. 2011. Opportunities of Interactive Learning Systems with Evolutions in Mobile Devices: A Case Study. In: *The 2011 International Conference on Internet Computing*, 18/21 July 2011, Las Vegas, USA. pp. 238-244

[6] Lu, J., Meng, Z., Lu, F. and Stav, J. B. 2010. A New Approach in Improving Operational Efficiency of Wireless Response System. In: *10th IEEE International Conference on Computer and Information Technology*, 29 Jun/1 July 2010, Bradford, UK. pp. 2676-2683

[7] Sawsaa, A., Lu, J. and Meng, Z., Using an Application of Mobile and Wireless Technology in Arabic Learning System, In: *International Conference on Education & Learning in Mobile Age 2011*, 1/2 June 2011, Lake District, UK. pp. 171-186

[8] Lu, J., Sundaram, A., Meng, Z., Priya, A., Lu, G. and Stav, J. B. 2011. MOBILE EXAM SYSTEM – MES: Architecture for Database Management System, In: *International Conference on Education & Learning in Mobile Age 2011*, 1-2 June 2011, Lake District, UK. pp. 1-20

REFERENCES

Abid, Z., Chabridon, S., Conan, D. 2009. A framework for quality of context management. In: *Proceedings of First International Workshop on Quality of Context*. 25-26 June, Stuttgart Germany. pp.120-131

Achilleos, A., Yang, K., Georgalas, N. 2010. Context modelling and a context-aware framework for pervasive service creation: a model driven approach. *Pervasive and Mobile Computing*. 6, pp. 281-296

Agostini, A., Bettini, C., Riboni, D., 2009. Hybrid reasoning in the CARE middleware for context-awareness. *International Journal of Web Engineering and Technology*. 5(1), pp. 3-23

Al-Hmouz, A., Shen, J., Al-Hmouz, R., Yan, J. 2012. Modeling and simulation of an Adaptive Neuro-Fuzzy Interface System (ANFIS) for mobile learning. *IEEE Transaction on Learning Technologies. 5 (3)*, pp. 226-237

Al-Sammarraie, M. H. 2011. *Policy-based approach for context-aware systems*. PhD Thesis, De Montfort University

Allen, F., Ambikairajah, E., Lovell, N., and Celler, B. 2006. An adaptive Gaussian mixture model approach to accelerometer-based movement classification using time-domain features, In: *Proceedings of the 28th EMBS Annual International Conference, 29-30 August, New York, USA*. pp. 3600-3603

Arnaboldi, V., Conti, M., Delmastro, F. 2013. CAMEO: A novel context-aware middleware for opportunistic mobile social networks. *Pervasive and Mobile Computing.* 11, pp.148-167

Anagnostopoulos, C. B., Tsounis, A., Hadjiefthymiades, S. 2007. Context awareness in mobile computing environments. *Wireless Personal Communications*. 42(3), pp. 445-464

Apple Inc. 2010. *iPhone – Technical Specifications*. [Online]. [Accessed July 2013]. Available from: http://support.apple.com/kb/sp2

Apple Inc. 2012a. *iPhone 3G –Technical Specifications*. [Online]. [Accessed July 2013]. Available form: http://support.apple.com/kb/sp495

Apple Inc. 2012b. *iPhone 3GS –Technical Specifications*. [Online]. [Accessed July 2013]. Available form: http://support.apple.com/kb/sp565

Apple Inc. 2012c. *iPhone 4 – Technical Specifications*. [Online]. [Accessed July 2013]. Available from: http://support.apple.com/kb/sp587

Apple Inc. 2013a. *iPhone 4S Tech Specs*. [Online]. [Accessed July 2013]. Available from: http://www.apple.com/uk/iphone-4s/specs/

Apple Inc. 2013b. *iPhone 5 – Technical Specifications*. [Online]. [Accessed July 2013]. Available from: http://support.apple.com/kb/sp655

Apple Inc. 2013c. *iPhone 5S*. [Online]. [Accessed Decmber 2013]. Available from: http://www.apple.com/uk/iphone-5s/specs/

Badidi, E., Esmahi, L. 2011. A cloud-based approach for context information provisioning. *World of Computer Science and Information Technology Journal*. 1(3), pp. 63-70

Bae, I.-H. 2014. An ontology-based approach to ADL recognition in smart homes. *Future Generation Computer Systems*. 33, pp.32-41

Bao, L., and Intille, S. S. 2004. Activity recognition from user-annotated accelerometer data. *Lecture Notes in Computer Science*. 3001, pp. 1-17

Barralon, P., Vuillerme, N., and Noury, N. 2006. Walk detection with a kinematic sensor: Frequency and wavelet comparison. In: *Proceedings of the 28th Annual International Conference of the IEEE Engineering in Medicine and Biology Society: Engineering Revolution in BioMedicine, 29-30 August, New York*, USA. pp.1711-1714

Basaeed, E. I., Berri, J., Zemerly, M. J., Benlamri, R. 2007. Web-based context-aware m-learning architecture. *International Journal of Interactive Mobile technologies*. 1(1), pp. 5-15

Baumer, E. P.S., Khovanskaya, V., Adams, P., Pollak, J. P. 2013. Designing for engaging experiences in mobile social-health support systems. *IEEE Pervasive Computing*. 12(3), pp. 32-39

Beamon, B., and Kumar, M. 2010. Adaptive context reasoning in pervasive systems, In: *Proceeding of the 9th International Workshop on Adaptive and Reflective Middleware, 29 Nov. – 03 Dec., Bangalore, India*. pp. 10-17

Bellavista, P., Corradi, A., Fanelli, M., Foschini, L. 2013. A survey of context data distribution for mobile ubiquitous systems. *ACM Computing Survey*. 44(4), pp. 24:1-25

Benabderrahmane, S., Smail-Tabbone, M., Poch, O., Napoli, A., Devignes, M.-D. 2010. IntelliGO: a new vector-based semantic similarity measure including annotation origin. *BMC Bioinformatics*. 11, pp. 1-16

Bettini, C., Brdiczka, O., Henricksen, K., Indulska, J., Nicklas, D., Ranganathan, A., Riboni, D. 2010. A survey of context modelling and reasoning techniques. *Pervasive and Mobile Computing*. 6, pp. 161-180

Bieber, G., Luthardt, A., Peter, C., and Urban, B. 2011. The hearing trousers pocket – activity recognition by alternative sensors. In: *Proceedings of the 4th International Conference on Pervasive Technologies Related to Assistive Environments, 25-27, Crete Greece*. [no pagination]

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

Bolchini, C., Curino, C., Schreiber, F. A., Tanca, L. 2006. Context integration for mobile data tailoring. In: *7th International Conference on Mobile Data Management, 10/12 May, Nara Japan.* [no pagination]

Bolchini, C., Curino, C.A., Orsi, G., Quintarelli, E., Rossato, R., Schreiber, F.A., Tanca, L. 2009. And what can context do for data? *Communications of the ACM*. 52(11), pp.136-140

Boldrini, C., Conti, M., Delmastro, F., Passarella, A. 2010. Context- and social-aware middleware for opportunistic networks. *Journal of Network and Computer Applications*. 33, pp.525-541

BOSCH, 2008. *BMP085 digital pressure sensor*. [Online]. [Accessed 01 July 2013]. Available from: http://www.adafruit.com/datasheets/BMP085_DataSheet_Rev.1.0_01July2008.pdf

Brezmes, T., Gorricho, J.-luis, and Cotrina J. 2009. Activity recognition from accelerometer data on a mobile phone. In: *Proceedings of the 10th International Work-Conference on Artificial Neural Networks, 10-12 June, Salamanca, Spain*. pp. 796-799

Buchholz, T., Kpper, A., Schiffers, M. 2003. Quality of Context: What It Is and Why We Need It? In: *Proceedings of the 10th International Workshop of the HP Open View University Association. 31 July -1 August, Geneva Switzerland*. [no pagination]

Burchfield, T. R., and Venkatesan, S. 2007. Accelerometer-based human abnormal movement detection in wireless sensor networks. In: *Proceedings of the 1st ACM SIGMOBILE International Workshop on System and Networking Support for Healthcare and Assisted Living Environments, 11-14 June, Puerto Rico, USA*. pp. 67-69

Campbell, A. T., Choudhury, T., Hu, S., Lu, H., Mukerjee, M. K., Rabbi, M., and Raizada, R. D. 2010. NeuroPhone: brain-mobile phone interface using a wireless EEG headset, In: *Proceedings of the Second ACM SIGCOMM Workshop on Networking, System, and Applications on Mobile Handhelds, 30 August, Delhi, India*. pp. 15-20

Carreras, A., Delgado, J., Rodriguez, E., Barbosa, V., Andrade, M. T., Arachchi, H. K., Gogan, S. Kondoz, A. M. 2010. A Platform for context-aware and digital rights management-enabled content adaptation. *IEEE Multimedia*. 17(2), pp. 74-89

Charlon, Y., Bourennane, W., Bettahar, F., Campo, E. 2013. Activity monitoring system for elderly in a context of smart home. *IRBM*. 34, pp. 60-63

Chernbumroong, S., Atkins, A. S., Yu, H. 2011. Activity classification using a single wrist-worn acceleroter. In: *5th International Conference on Software, Knowledge Information, Industrial Management and Applications, 8-11 September, Benevento, Italy*. pp. 21-25

Chen, G. 2004. *Solar: Building a context fusion network for pervasive computing*. PhD Thesis, Dartmouth College

Chen, G., Kotz, D. 2000. *A survey of context-aware mobile computing research*. Technical report, Dartmouth College

Chen, H. 2003. *An intelligent broker architecture for context-aware systems.* PhD Thesis, University of Maryland

Chen, J., Kwong, K., Chang, D., Luk, J. and Bajcsy, R. 2005. Wearable sensors for reliable fall detection. In: *Proceedings of the IEEE Engineering in medicine and Biology Conference, 1-4 September, Shanghai, China*. pp. 3551-3554

Chen, P. and Zhi, X. 2011. Smart home architecture based on event-driven DPWS. *Journal of Shanghai University*. 15(5), pp. 386-390

Cheng, S.-T., Wang, C.-H., Horng, G.-J. 2012. OSGi-based smart home architecture for heterogeneous network. *Expert Systems with Applications*. 39, pp. 12418-12429

Chon, Y., Cha, H. 2011. LifeMap: a smartphone based context provider for location-based service. *IEEE Pervasive Computing*. 10(2), pp. 58-67

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

Chrofi, H. O., Sevkli, A. Z., Bousbahi, F. 2012. Mobile learning adaptation through a device based reasoning. *Procedia – Social and Behavioral Sciences*. 47, pp.1707-1712

Chung, W.-Y., Oh, S.-J. 2006. Remote monitoring system with wireless sensors module for room environment. *Sensors and Actuators B*. 113, pp. 64-70

Coppola, P., Mea, V. D., Gaspero, L. D., Menegon, D., Mischis, D., Mizzaro, S., Scagnetto, I., and Vassena, L. 2010. The Context-aware browser. *IEEE Intelligent Systems*. 25(1), pp. 38-47

Costa, P. D. 2003. *Towards a services platform for context-aware applications*. MSc Thesis, University of Twente

Covington, M. J., Moyer, M. J., Ahamad, M. 2000. *Generalized role-based access control of securing future applications*. Technical Report GIT-CC-00-02, Georgia Institute of Technology

Dai, J., Bai, X., Yang, Z., Shen, Z., Xuan, D. 2010. Mobile phone-based pervasive fall detection. *Journal of Personal and Ubiquitous Computing*. 14(7), pp. 633-643

Daniele, L. M. 2006. *Towards a Rule-based Approach for Context-Aware Applications*. MSc. Dissertation, University of Cagliari

Denning, T., Andrew, A., Chaudhri, R., Hartung, C., Lester, J., Borriello, G., and Duncan, G. 2009. Balance: towards a usable pervasive wellness applications with accurate activity inference. In: *Proceedings of the 10th Workshop on Mobile Computing System and Applications, 23-24 February, Santa Cruz, USA*. 5, pp. 1-6

Dey, A. K. 2001. Understanding and using context. *Personal Ubiquitous Computing*. 5(1), pp.4-7

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, pp. 97-166

Dey, A. K., Salber, D., Futakawa, M. 1999. An Architecture to Support Context-Aware Applications. In: *12th Annual ACM Symposium on User Interface Software and Technology, 7-10 November, Asheville, USA*. [no pagination]

Dey, A. K. 2000. *Providing Architectural Support for Building Context-Aware Applications*. Ph.D Thesis, Georgia Institute of Technology

Dey, A. K. and Abowd, G. D. 1999. Towards a better understanding of context and context-awareness. Atlanta: Georgia Institute of Technology

Ding, D., Cooper, R. A., Pasquina, P., Fici-pasquina, L. 2011. Sensor technology for smart homes. *Maturitas*. 69, pp.131-136

Dinh, C., and Struck, M. 2009. A new real-time fall detection approach using fuzzy logic and neural network. In: *6th International Workshop on Wearable Micro and Nano Technologies for Personalized Health, 24-26 June, Oslo, Norway*. pp. 57-60

DrRobot Inc., 2005. *DHM5150 human motion sensor module user manual*. [Online]. [Accessed 01 July 2013]. Available from: http://www.drrobot.com/products/item_downloads/DHM5150_1.pdf

Duan, Q., Miao, D., Zhang, H., Zheng, J. 2007. Personalised Web retrieval based on Rough-Fuzzy method. *Journal of Computational Information System*. 3(2), pp. 203-208

ElecFreaks, 2011. *Specification of DYP-ME003*. [Online]. [Accessed 01 July 2013]. Available from:http://elecfreaks.com/store/download/datasheet/sensor/DYP-ME003/Specification.pdf

ElecFreaks, 2013. *New product post43 - Infrared sensor and speed module*. [Online]. [Accessed 01 July 2013]. Available from: http://www.elecfreaks.com/4360.html

Elecrow, 2013. *HR202 humidity sensor module*. [Online]. [Accessed 01 July 2013]. Available from: http://www.elecrow.com/sensors-c-111/environment-c-111_112/hr202-humidity-sensor-module-p-674.html

Emartee, 2013. *Arduino mini sound sensor*. [Online]. [Accessed 01 July 2013]. Available from:http://www.emartee.com/product/42148/Mini%20Sound%20Sensor%20%20Arduino%20Compatible

Espada, J. P., Crespo, R. G., Martinez, O. S., G-Bustelo, B. C. P., Lovelle, J. M. C. 2012. Extensible architecture for context-aware mobile web applications. *Expert System with Applications*. 39, pp. 9686-9694

Feki, M. A., Kawsar, F., Boussard, M., Trappeniers, L. 2013. The Internet of Things: the next technological revolution. *Computer*. 46(2), pp. 24-25

Filho, J. B., Martin, H. 2008. A quality-aware context-based access control model for ubiquitous applications. In: *3rd International Conference on Digital Information Management, 13-16 November, London, UK*. pp. 113-118

Filho, J. B., Miron, A. D., Satoh, I. 2010. Modelling and measuring quality of context information in pervasive environment. In: *24th IEEE International Conference on Advanced Information Networking and Applications, 20-23 April, Perth, Australia*. pp. 690-697

Fong, A. C. M., Zhou, B., Hui, S. C., Hong, G. Y., Do, T. A. 2011. Web content recommender system based on consumer behaviour modelling. *IEEE Transactions on Consumer Electronics*. 57(2), pp. 962-969

Ganesan, P., Garcia-Molina, H., Widom, J. 2003. Exploiting hierarchical domain structure to compute similarity. *ACM Transactions on Information System*. 21, pp. 64-93

Gao, C., Kong, F., and Tan, J. 2009. Healthaware: Tackling obesity with health aware smart phone system. *In: 2009 IEEE International Conference on Robotics and Biomimetics, 18-22 December, Guilin, China*. pp.1549-1554

Gavalas, D., Economou, D. 2011. Development platform for mobile applications: status and trends. *IEEE Software*. 28(1). pp. 77-86

Gossweiler, R., McDonough, C., Lin, J., Wang, R. 2011. Argos: Building a Web-Centric Application Platform on Top of Android. *IEEE Pervasive Computing*. 10(4), pp. 10-14

Gray, P. D., Salber, D. 2001. Modelling and using sensed context information in the design of interactive adaptation. In: *Proceedings of the 8th IFIP International Conference on Engineering for Human-Computer Interaction, 11-13 May, Toronto Canada*. pp. 317-336

Gu, T., Pung, H. K., Zhang, D. Q. 2005. A service-oriented middleware for building context-aware service. *Journal of Network and Computer Applications*. 28, pp. 1-18

Gu, T., Pung, H. K., Zhang, D. Q. 2004. Toward an OSGi-Based Infrastructure for Context-Aware Applications. *IEEE Pervasive Computing*. 3(4), pp. 66-74

Guan, D., Yuan, W., Gavrilov, A., Lee, S., Lee, Y., Han, S. 2006. Using fuzzy decision tree to handle uncertainty in context deduction. In: *Proceedings of the 2006 international conference on Intelligent computing, 16-19 August, Kunming China*. pp: 63-72

Grønli, T.-M., Hansen, J., Ghinea, G. 2010. A context-aware meeting room. In: *34th Annual IEEE Computer Software and Applications Conference, 19-23 July, Seoul, Korea*. pp. 311-316

Grønli, T.-M. 2012. *Cloud computing and context-aware – a study of the adapted user experience*. PhD Thesis, Brunel University

Gruber, T. R., 1993. A translation approach to portable ontology specification. *Knowledge Acquisition*. 5(2), pp.199–220

GSMArena, 2013a. *HTC Google Nexus One*. [Online]. [Accessed July 2013]. Available from: http://www.gsmarena.com/htc_google_nexus_one-3069.php

GSMArena, 2013b. *Samsung Google Nexus S*. [Online]. [Accessed July 2013]. Available from: http://www.gsmarena.com/samsung_google_nexus_s-3620.php

GSMArena, 2013c. *LG Nexus 4*. [Online]. [Accessed July 2013]. Available from: http://www.gsmarena.com/lg_nexus_4_e960-5048.php

GSMArena, 2013d. *ASUS Google Nexus 7*. [Online]. [Accessed July 2013]. Available from: http://www.gsmarena.com/asus_google_nexus_7-4850.php

GSMArena, 2013e. *Google Nexus 10*. [Online]. [Accessed July 2013]. Available from: http://www.phonearena.com/phones/Google-Nexus-10_id7551

GSMArena, 2013f. *LG Nexus 5*. [Online]. [Accessed July 2013]. Available from: http://www.gsmarena.com/lg_nexus_5-5705.php

Győrbíró, N., Fábián, Á., Hományi, G. 2009. An activity recognition system for mobile phones. *Mobile Networks and Applications*. 14(1), pp. 82-91

Hansen, T. R., Eklund, J. M., Sprinkle, J., Bajcsy, R., and Sastry S. 2005. Using smart sensors and a camera phone to detect and verify the fall of elderly persons, In: *European Medicine, Biology, and Engineering Conference, 20-25 November, Prague, Czech Republic*. [no pagination]

He, Z., Jin, L., Zhen, L., and Huang, J. 2008. Gesture recognition based on 3D accelerometer for cell phones interaction. In: *IEEE Asia Pacific Conference on Circuits and Systems, 30 November - 3 December, Macau*. pp. 217-220

Heinz, E., Kunze, K., Gruber, M., Bannach, D., and Lukowicz, P. 2006. Using wearable sensors for Real-time recognition tasks in games of martial arts – an initial experiment, In: *2006 IEEE Symposium on Computational Intelligence and Games, 22-24 May, Reno USA*. pp. 98-102

Henricksen, K. 2003. *A framework for context-aware pervasive computing applications.* PhD Thesis, The University of Queensland

Hevner, A. R., March, S. T., Park, J. 2004. Design science in information systems research. *MIS Quarterly*. 28(1), pp.75-105

Hoareau, C., Satoh, I. 2009. Modelling and processing information for context-aware computing: a survey. *New Generation Computing*. 27, pp.177-196

Hornbæk, K. 2006. Current practice in measuring usability: challenges to usability studies and research. *International Journal of Human-Computer Studies*. 64, pp. 79-102

Hong, J., Suh, E.-H., Kim, J., Kim, S. 2009. Context-aware system for proactive personalized service based on context history. *Expert Systems with Applications*. 36, pp. 7448-7457

Hosseini-Pozveh, M., Nematbakhsh, M., Movahhedinia, N. 2009. A multidimensional approach for context-aware recommendation in mobile commerce. *International Journal of Computer Science and Information Society*. 3(1), [no pagination]

Hristova, A. 2008. *Conceptualization and design of a context-aware platform user-centric applications*. MSc. Dissertation, Norwegian University of Science and Technology

Hsu, I-C. 2011. An architecture of mobile Web 2.0 context-aware applications in ubiquitous Web. *Journal of Software*. 6(4), pp.705-715

Huang, J., Dang, J., Huhns, M. N., Shao, Y. 2005. Ontology alignment as a basis for mobile service integration and invocation. *J. Pervasive Comput. & Comm.* 1(2), pp. 1-11

Huang, Y.-M., Kuo, Y.-H., Lin, Y.-T., Cheng, S.-C. 2008. Toward interactive mobile synchronous learning environment with context-awareness service. *Computers & Education*. 50, pp.1205-1226

Huang, Z., Lu, X., and Duan, H. 2011. Context-aware recommendation using rough set model and collaborative filtering. *Artificial Intelligence Review*. 35(1), pp. 85-99

Huynh, D. T. G. 2008. *Human activity recognition with wearable sensors*. PhD Thesis, Technische Universität Darmstadt

Jafari, R., Li, W., Bajcsy, R., Glaser, S., and Sastry, S. 2007. Physical activity monitoring for assisted living at home. In: *Proceedings of 4th International Workshop on Wearable and Implantable Body Sensor Networks, 26-28 March, Aachen Germany*. pp. 213-219

Jantunen, I., Laine, H., Huuskonen, P., Trossen, D., Ermolov, V.. 2008. Smart Sensor Architecture for mobile-terminal-centric ambient intelligence. *Sensors and Actuators A: Physical*. 142, pp.352-360

Jarvinen, P., Jarvinen. T., Lahteenmaki, L., and Sodergard, C. 2008. Hyperfit: hybrid media in personal nutrition and exercise management. In: *2nd International Conference on Pervasive Computing Technologies for Healthcare, 30 January-2 February, Tampere, Finland*. pp. 222-226

Jiang, S., Cao, Y., Iyengar, S., Kuryloski, P., Jafari, R., Xue, Y., Bajcsy, R., and Wicker, S. 2008. CareNet: an integrated wireless sensor networking environment for remote healthcare. In: *Proceedings of the ICST 3rd International Conference on Body Area Networks, 13-15 March, Anzona, USA*. [no pagination]

Könönen, V., and Mäntyjärv, J. 2008. Decision making strategies for mobile collaborative context recognition. In: *Proceedings of the 4th International Mobile Multimedia Communications Conference, 7-9 July, Oulu, Finland*. [no pagination]

Karantonis, D., Narayan an, M., Marthie, M., Lovell, N., and Celler, B. 2006. Implementation of a real-time human movement classifier using a tri-axial accelerometer for ambulatory monitoring. *IEEE Transactions on Information Technology in Biomedicine*. 10(1), pp. 156-167

Kasaki, N., Kurabayashi, S., Kiyoki, Y. 2012. A geo-location context-aware mobile learning system with adaptive correlation computing methods. *Procdia Computing Science*. 10, pp. 593-600

Khan, A. M., Lee, Y., and Lee, S. Y. 2010. Human activity recognition via an accelerometer-enabled-smartphone using kernel discrimination analysis. In: *5th International Conference on Future Information Technology, 20-24 May, Busan, Korea*. pp. 1-6

Khan, W. Z., Xiang, Y., Aalsalem, M. Y, Arshad, Q. 2012. Mobile sensing system: a survey. *IEEE Communications Survey & Tutorials*. 15(1), pp. 402-427

Kim, H.-M., Lee, K.-H. 2006. Device independent web browsing with CC/PP annotation. *Interacting with Computers*, 18(2). pp. 283-303

Kim, Y., Lee, K. 2006. A quality measurement method of context information in ubiquitous environments. In: *2006 International Conference on hybrid Information Technology, 9-11 November, Cheju Island, Korea*. pp. 576-581

Korber, H.-J. Wattar, H., Scholl, G. 2007. Modular wireless real-time sensor/actuator network for factory automation applications. *IEEE Transactions on Industrial Informatics*. 3(2), pp.111-119

Koskela, T., Kostamo, N., Kassinen, O., Ohtonen, J., Ylianttila, M. 2007. Towards context-aware mobile Web 2.0 service architecture. In: *International Conference on Mobile Ubiquitous Computing, Systems, Services and Technologies, 4-9 November, Papeete, French Polynesia*. pp. 41-48

Krause, E. F. 1987. *Taxicab Geometry: An Adventure in Non-Euclidean Geometry*. New York: Dover Publications.

Lane, N. D., Miluzzo, E., Lu, H., Peebles, D., Choudhury, T., and Campbell, A. T. 2010. A survey of mobile phone sensing. *IEEE Communications Magazine*. 48(9), pp. 140-150

Langheinrich, M. 2001. Privacy by design – principles of privacy aware ubiquitous systems. In: *UBICOM 2001, LNCS 2201*. pp. 237-271

Lanir, J., Kuflik, T., Wecker, A. J., Stock, O., Zancanaro, M. 2011. Examining proactiveness and choice in a location-aware mobile museum guide. *Interacting with Computers*. 23, pp. 513-524

Lee, H.J., Park, S.J. 2007. MONERS: a news recommender for the mobile Web. *Expert Systems with Applications*. 32 (1), pp.143–150

Lee, H. Y., Ahn, H., and Han, I. 2007. VCR: Virtual community recommender using the technology acceptance model and user’s needs type. *Expert Systems with Applications*. 33(4), pp. 984-995

Lee, Y.-S., and Cho, S.-B. 2011. Activity recognition using Hierarchical Hidden Markov Models on smartphone with 3D accelerometer. In: *Proceedings of the 6th International Conference on Hybrid Artificial Intelligent Systems, 23-25 May, Wroclaw, Poland*. pp. 460-467

Lenzerini, M. 2002. Data integration: a theoretical perspective. *In: Proceeding of 21st ACM SIGMOD-SIGACT-SIGART symposium on Principles of Database System, 3-6 June, Madison, USA*. pp. 233-246

Liang, L., Huang, L., Jiang, X., Yao, Y. 2008. Design and implementation of wireless smart-hone sensor network based on ZigBee protocol. In: *International Conference on Communications, Circuits and Systems, 25-27 May, Xiamen, China*. pp. 434-438

Lin, R.-T., Hsu, C.-S., Chun, T.Y., Cheng, S.-T. 2008. OSGi-based smart home architecture for heterogeneous network. In: *3rd International Conference on Sensing Technology, 30 November-3 December, Tainan, Taiwan*. pp. 527-532

Liptchinsky, V., Khazankin, R., Schulte, S., Satzger, B., Truong, H.-L., Dustdar, S. 2014. On modelling context-aware social collaboration process. *Information System*. 43, pp.66-82

Lombriser, C., Bharatula, N. B., Roggen, D., and Troster, G. 2007. On-body activity recognition in a dynamic sensor network. In: *Proceedings of the ICST 2nd International Conference on Body Area Network, 11-13 June, Florence, Italy*. pp. 1-6

Lu, H., Pan, W., Lane, N. D., Choudhury, T., and Campbell, A. T. 2009. SoundSense: scalable sound sensing for people-centric applications on mobile phones. In: *Proceedings of the 7th*

International Conference on Mobile systems, Applications, and Services, *22-25 June, Wroclaw, Poland*. pp. 165-178

Lu, J., Meng, Z., Lu, F. and Stav, J. B. 2010. A New Approach in Improving Operational Efficiency of Wireless Response System. In: *10th IEEE International Conference on Computer and Information Technology, 29 Jun - 1 July 2010, Bradford, UK*. pp. 2676-2683

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

Lumpkin, W. 2013. The Internet of Things meets cloud computing. *IEEE Consumer Electronics Magazine*. 2(2), pp. 47-51

Manzoor, A. 2010. *Quality of context in pervasive systems: models, techniques, and applications*. PhD Thesis, Vienna University of Technology

Manzoor, A., Truong, H.-L., Dustdar, S. 2008. On the evaluation of quality of context. In: *Proceedings of the 3rd European Conference on Smart Sensing and Context. 29-31 October 2008, Zurich, Switzerland*. pp. 140-153

Markris, P., Skoutas, D. N., Skianis, C. 2013. A survey on context-aware mobile and wireless networking: on networking and computing environments' integration. *IEEE Communications Survey & Tutorials*. 15(1), pp. 362-368

Martin, E., 2011. Enhancing context awareness with activity recognition and radio fingerprinting. In: *5th IEEE International Conference on Semantic Computing, 19-21 September, Palo Alto, USA*. pp. 263-266

Medjahed, H., Istrate, D., Boudy, J., Baldinger, J.-L., Dorizzi, B. 2011. A pervasive multi-sensor data fusion for smart home healthcare monitoring. In: *2011 IEEE International Conference on Fuzzy Systems, 27-30, June, Taipei. Taiwan*. pp. 1466-1473

Miluzzo, E., Cornnelius, C. T., Ramaswamy, A., Choudhury, T., Liu, Z., Campbell, A. T. 2010a. Darwin phones: the evolution of sensing and inference on mobile phones. In: *Proceedings of the 8th International Conference on Mobile Systems, Applications, and Service*, *15-18 June, San Francisco, USA*. pp. 5-20

Miluzzo, E., Wang, T., and Campbell A. T. 2010b. EyePhone: activating mobile phones with your eyes. *In: Proceedings of the Second ACM SIGCOMM Workshop on Networking Systems, and Applications, on Mobile Handhelds, 30 August, New Delhi, India*. pp. 15-20

Miluzzo, E., Papandrea, M., Lane, N. D., Lu, H., and Campbell, A. T. 2010c. Pocket, Bag, Hand, etc. – Automatically detecting phone context through discovery. In: *Proceedings of International Workshop on Sensing for App Phones, 2 November, Zurich, Switzerland*. [no pagination]

Musumba, G. W., Nyongesa, H. O. 2013. Context awareness in mobile computing: a review. *International Journal of Machine Learning and Applications*. 2(1), [no pagination]

Nickel, C., Busch, C., Rangarajan, S., and Mobius, M. 2011. Using- Hidden Markov Models for accelerometer-based biometric gait recognition. In: *IEEE 7th International Colloquium on Signal Processing and its Applications, 4-6 March, Pulau Pinang, Malaysia*. pp. 58-63

Notles, J. 2008. An architecture for context-aware mobile Web browsing. In: *9th Twente Student Conference on IT, 23 June, Enschede, Netherlands*. [no pagination]

Orlikowski, W. J. and Baroudi, J. J. 1991. Studying information technology research approaches and assumptions. *Information Science Research*. 2(1), pp.1-28

Padovitz, A., Loke, S. W., Zaslavsky, A. 2008. The ECORA framework: A hybrid architecture for context-oriented pervasive computing. *Pervasive and Mobile Computing*. 4, pp. 182-215

Pappas, I., Keller, T., Mangold, S., Popovic, M., Dietz, V., and Morari, M. 2004. A reliable gyroscope-based gait-phase detection sensor embedded in a shoe insole. *IEEE Sensor Journal*. 4(2), pp. 268-274

Parallax Inc. 2006. *Ultrasonic Distance Sensor*. [Leaflet]. California: Parallax Inc.

Parra, J., Hossain, M. A., Uribarren, A., Jacob, E., Saddik, A. E.. 2009. Flexible smart home architecture using device profile for Web service: a Peer-to-Peer approach. *International Journal of Smart Home*. 3(2), pp. 39-56

Pawlak, Z. 1982. Rough Sets. *International Journal of Computer & Information Sciences*. 11(5), pp.341-356

Pawkak, Z. and Skowron, A. 2007. Rough sets: some extensions. *Information Science*. 177, pp.28-40

Perez, A. J., Labrador, M. A., Barbeau, S. J. 2010. G-Sense: a scalable architecture for global sensing and monitoring. *IEEE Network*. 24(4), pp. 57-64

Pololu, 2013. *Flammable gas & smoke sensor*. [Online]. [Accessed 01 July 2013]. Available from:http://www.pololu.com/product/1480

Poveda-Villalón, M., Suárez-Figueroa, M. C., García-Castro, R., Gómez-Pérez, A. 2010. A context ontology for mobile environments. *In: Workshop on Context, Information and Ontologies - CIAO 2010 Co-located with EKAW 2010, 11-15 October, Lisbon, Portugal*. [no pagination]

Preece, J., Rogers, Y., and Sharp, H. 2002. Interaction design, beyond human-computer interaction. *John Wiley & Sons Inc*.

Preuveneers, D., Bergh, J. V., et al. 2004. Towards an extensible context ontology for ambient Intelligence. In: *Proceedings of the 2nd European Symposium on Ambient Intelligence, 8-11 November, Eindhoven, Netherland*. pp. 148-159

Preuveneers, D., Berbers, Y. 2006. Quality extension and uncertainty handling for context ontologies. In: *Proceedings of Context and Ontologies: Theory Practice and Applications, 28 August, Riva del Garda, Italy*. pp. 62-64

Pung, H. K., Gu, T., Xue, W., Palmes, P. P., Zhu, J., Ng, W. L., Tang, C. W., Chung, N. H. 2009. Context-aware middleware for pervasive elderly homecare. *IEEE Journal on Selected Area in Communications*. 27(4), pp. 510-524

Raento, M., Oulasvirta, A., Petit, R., Toivonen. 2005. ContextPhone: a prototyping platform for context-aware mobile applications. *IEEE Pervasive Computing*. 4(2). pp.51-59

Ram, S. 2004. *A lesson on data representation with XML*. [Online]. [Accessed 1 October 2013]. Available from: http://userpage.fu-berlin.de/~ram/pub/pub_jf47ht81Ht/xml_data_representation_en

Rarău, A., Pusztai, K., Salomine, I. 2005. MultiFacet item based context-aware applications. *International Journal of Computing & Information Science*. 3(2), pp. 10-18

Rarău, A. 2006. Multifact paradigm – support for context-aware application development. *PhD Thesis*, Technical University of Cluj-Napoca

Ravi, N., Dandekar. N., Mysore, P., and Littman, M. L. 2005. Activity recognition from accelerometer data. In: *Proceedings of the 17th Conference on Innovative Applications of Artificial Intelligence (IAAI), 22-26 June, Vancouver, British Columbia*. pp. 1541-1546

Rehman, K., Stajano, K., Coulouris, G. 2007. An architecture for interactive context-aware applications. *IEEE Pervasive Computing*. 6(1), pp. 73-80

Ricco, F., and Nguyen, Q. N. 2007. Acquisition and revising preferences in a critique-based mobile recommender system. *IEEE Intelligent Systems*. 22(3), pp. 22-29

Rohm Co., 2010. *Digital 16bit sensor output type ambient light sensor IC*. [Online]. [Accessed 01 July 2013]. Available from: http://www.elechouse.com/elechouse/images/product/Digital%20light%20Sensor/bh1750fvi-e.pdf

Romero, D., Rouvoy, R., Seinturier, L., Loiret, F. 2010. Integration of heterogeneous context resource in ubiquitous environments. In: *36th EUROMICRO Conference on Software Engineering and Advanced Applications, 1-3 September, Lille, France*. pp. 123-126

Roving Networks. 2011a. *Wifly serial adaptor - user manual*. [Online]. [Accessed 12 July 2013]. Available from: http://www.rovingnetworks.com/products/RN370

Roving Networks. 2011b. *Bluetooth wireless adaptor - user manual*. [Online]. [Accessed 12 July 2013]. Available from: http://www.rovingnetworks.com/products/RN274

Saeedi, S., El-Sheimy, N., Malek M.R., and Samany, N. N. 2010. An ontology based context modeling approach for mobile touring and navigation system. In: *Canadian Geomatics Conference and Symposium of Commission I, ISPRS, 15-18 June, Calgary, Canada*. [no pagination]

Sahafipour, F., Javidan, R. 2012. A comparative study of context modelling approaches and applying in an infrastructure. *Canadian Journal of Data Information and Knowledge Engineering*. 3(1), pp. 1-6

Salden, A., Beijnum, B.-J. V., Widya, I. 2004. *Quality of context modelling – A prerequisite for viable ad-hoc context-aware service provisioning*. [Online]. [Accessed 12 July 2013]. Available from: https://doc.novay.nl/dsweb/Get/Document-47467

Saponas, T. S., Lester, J., Froehlich, J., Fogarty, J., and Landay, J. 2008. *iLearn on the iPhone: Real-time human activity classification on commodity mobile phones*. Techical Report UW-CSE-08-04-02, University of Washington

Schäfer, R. 2001. Rules for using Multi-Attribute Utility Theory estimating a user's interests. In: Workshop Adaptivity and User Modelling, [no pagination]

Schilit, W. N. 1995. *A system architecture for context-aware mobile computing*. Ph.D thesis, Columbia University

Schilit, B. and Theimer, M. 1994. Disseminating active map information to mobile hosts. *IEEE Network*. 8(5), pp.22-32

Schilit, B. 1994. Context-aware computing applications. In: *Proceedings of the 1994 1st Workshop on Mobile Computing Systems and Applications, 8-9 December, 1994, California, USA*. pp. 85-90

Schmidt, A. 2002. *Ubiquitous computing – computing in context*. Ph.D Thesis, Lancaster University

Schmohl, R., and Baumgarten, U. 2008a. A generalized context-aware architecture in heterogeneous mobile computing environments. In: *Proceedings of the 4th International Conference on Wireless and Mobile Communications, 27 July – 1 August, Athens, Greece*. pp.118-124

Scott, K., and Benlamri, R. 2010. Context-aware service for smart learning space. *IEEE Transactions on Learning Technologies*. 3(3), pp.214-227

Sekkas, O., Anagnostopoulos, C. B., Hadjiefthymiades, S. 2007. Context fusion through imprecise reasoning. *IEEE International Conference on Pervasive Service, 15-20 July, Istanbul, Turkey*. pp. 88-91

Sha, K., Zhan, G., Shi, W., Lumley, M., Wiholm, C., Arnetz, B. 2008. SPA: a smart phone assisted chronic illness self-management system with participatory sensing. In: *Proceedings of the 2nd International Workshop on System and Networking Support for Health Care and Assisted Living Environments, 17 June, Breckenrige, USA*. [no pagination]

Sheikh, K., Wegdam, M., and Sinderen, M. V. 2008. Quality-of-context and its use for protecting privacy in context aware system. *Journal of Software*. 3(3), pp. 83-93

Shirazi, A., S., Winkler, C., Schmidt, A. 2010. SENSE-SATION: An extensible platform for integration of phones into the web. In: *2010 Internet of Things, 29 November-1 December, Tokyo, Japan*. pp. 1-8

Shrestha, A. 2010. MobileSOA framework for context-aware mobile applications. In: 7th *International Conference on Mobile Data Management, 10-12 May, Nara Japan*. pp. 297-298

Siewiorek, D., Smailagic, A., Furukawa, J., Krause, A., Moraveji, N., Reiger, K., Shaffer, J., Wong, F. L. 2003. SenSay: a context-aware mobile phone. In: *Proceedings of 7th IEEE International Symposium on Wearable Computers, 21-23 Oct., New York, USA*. pp. 248-257

Simpson, J. A., Weiner, E. S. C. 1991. *The Oxford English Dictionary*. 2nd Ed. Oxford: Clarendon Press.

Sixsmith, A., and Johnson, N. 2004. A smart sensor to detect the fall of the elderly. *IEEE Pervasive Computing*. 3(2), pp. 42-47

Smailagic, A., Siewiorek, D. P., Anhalt, J., Kogan, D., Wang, Y. 2001. Location sensing and privacy in a context-aware computing environment. *IEEE Wireless Communications*. 9, pp. 10-17

Strang, T., Linnhoff-Popien, C. 2004. A context modelling survey. In: *UbiComp 1st International Workshop on Advanced Context Modelling, Reasoning, and Management, 7-10 September 2004, Nottingham, UK*. pp. 34-41

Sun, L., Zhang, D., Li, B., Guo, B., and Li, S. 2010. Activity recognition on an accelerometer embedded mobile phone with varying positions and orientations. In: *Proceedings of 7th Internatonal Conference on Ubiquitous Intelligence and Computing, 26-29 October, Xi'an China*. pp. 548-562

Suresh, G. V., Reddy, E. V., Reddy, E. S. 2012. Uncertainty data classification using rough set theory. In: *Proceedings of the International Conference on Information System Design and Intelligent Applications 2012, 5-7 January, Visakhapatnam India*. pp. 869-877

TalebiFard, P., Leung, V. C.M. 2011. A data fusion approach to context-aware service delivery in heterogeneous network environments. *Procedia Computer Science*. 5, pp. 312-319

Tesoriero, R., Gallud, J. A., Lozano, M. D., Penichet, V. M.R. 2010. CAUCE: Model-driven development of context-aware applications for ubiquitous computing environments. *Journal of Universal Computer Science*, 16(15), pp. 2111-2138

Toye, E., Madhavapeddy, A., Sharp, R., Scott, D., Blackwell, A., Upton, E. 2004. Using camera-phones to interactive with context-aware mobile service. *Technical reports published by the University of Cambridge Computer Laboratory*, 609, pp. 1-23

Vaishnavi, V., and Kuechler, B. 2004. Design science research in information system. *Association for Information Systems*. [Online]. [Accessed 28 March 2013]. Available from: http://desrist.org/design-research-in-information-systems/.

Waluyo, A. B., Yeoh, W.-S., Pek, I., Yong, Y., and Chen, X. 2010. MobiSense: mobile body sensor network for ambulatory monitoring. *ACM Transactions on Embedded Computing Systems.*10(1), pp. 13-32

Wang, X., Rosenblum, D., Wang, Y. 2012. Context-aware mobile music recommendation for daily activities. In*: Proceedings of the 20th ACM International Conference on Multimedia, 29 October - 2 November, Nara Japan*. pp. 99-108

Ward, J. A., Bharatula, N. B., Tröster, G., Lukowicz, P. 2006. *Continuous activity recognition in the kitchen using miniaturized sensor button*. Internal technical notes, University of Lancaster

Weiser, M. 1991. The computer for 21st century. *Scientific American*. 265(3), pp. 94-104

Wieland, M., Käppeler, U.-P., Levi, P., Leymann, F., and Nicklas, D. 2009. Towards integration of uncertain sensor data into context-aware workflows. *GI Jahrestagung*. pp. 2029-2040

Wu, C.-L., Liao, C.-F., Fu, L.-C. 2007. Service-oriented smart-home architecture based on OSGi and mobile-agent technology. *IEEE Transactions on Systems, Man, and Cybernetics – Part C: Applications and Review*. 37(2), pp.193-205

Wu, W. H., Bui, A. A., Batalin, M. A., Au, L. K., Binney, J. D., and Kaiser, W. J. 2008. Medic: medical embedded device for individualized care. *Artificial Intelligence in Medicine*. 42(2), pp. 137-152

Xia, F., Tian, Y.-C., Li, Y., Sung, Y. 2007. Wireless sensor/actuator network design for mobile control applications. *Sensors*. 7(10), pp. 2157-2173

Yang, A. Y., Jafari, R., Sastry, S. S., and Bajcsy, R. 2009. Distributed recognition of human actions using wearable motion sensor networks. *Journal of Ambient Intelligence and Smart Environments*. 1, pp.1-13

Yang, J., Chen, Y., Lee, G., Liou, S. and Wang, J. 2007. Activity recognition using one triaxial accelerometer: a Neuro-fuzzy classifier with feature reduction. In: *Proceedings of the 6th International Conference on Entertainment Computing, 15-17 September, Shanghai, China*. pp. 395-400

Yap, G.-E., Tan, A.-H., Pang, H.-H. 2007. Discovering and exploiting causal dependencies for robust mobile context-aware recommenders. *IEEE Transactions on Knowledge and Data Engineering*. 19(7), pp. 977-992

Yu, Z., Zhou, X., Zhang, D., Chin, C.-Y., Wang, X., Men, J. 2006. Supporting context-aware media recommendation for smart phones. *IEEE Pervasive Computing*. 5(3), pp. 68-75

Yue, W. N., Wang, Y., Wang, G. P., Wang, H., Dong, S. H.. 2005. Architecture of intelligent interaction system based on context awareness. *Journal of Computer-aided Design & Computer Graphics*. 17(1), pp. 74-79

Zeng, Y., Li, D., Vasilakos, A. V. 2013. Real-time data report and task execution in wireless sensor and actuator networks using self-aware mobile actuators. *Computer communications*. 36, pp. 988-997

Zhang, J., Qi, Y., Hou, D., Li, M. 2009. A table-driven programming paradigm for context-aware application development. In: *2009 9th Annual International Symposium on Applications and Internet, 20-24 July, Washington, USA*. pp. 121-124

Zhang, X., Yu, Z., Tian, J., Wang, Z., Guo, B. 2012. Context-aware mobile web browsing based on HTML5. In: *9th International Conference on Ubiquitous Intelligence and Computing and 9th International Conference on Automatic and Trusted Computing, 4-7 September, Fukuoka, Japan*. pp. 945-950

Zhuang, J., Mei, T., Hoi, S., Xu, Y.-Q., Li, S. 2011. When recommendation meets mobile: contextual and personalized recommendation on the go. In: *Proceedings of ACM Ubicomp, 17-21 September, Beijing, China*. pp. 945-950

Zimmer, T. 2006. Quality of context – handling context dependencies. In: *2nd International Workshop on Personalized Context Modelling and Management for Ubiquitous Applications*, *18 September, Califorina, USA*. [no pagination]

Zimmermann, A., Lorenz, A., Oppermann, R. 2007. An operational definition of context. *Lecture Notes in Computer Science*. 4635, pp. 558-571

ZXing Group. 2013. *ZXing – Multi-format 1D/2D barcode image processing library with clients for Android, Java*. [Online]. [Accessed 12 July 2013]. Available from: http://code.google.com/p/zxing

APPENDIX

1. Context Acquisition and Command Execution with MCU

(1) Context Acquisition and Context Update

```
#include <at89x52.h>
#include <math.h>
#include "string.h"

#define   uchar unsigned char
#define   uint unsigned int
#define   SlaveAddress  0x46

sbit  sensor1 = P2^0; //Sound
sbit  sensor2 = P2^1; //Human
sbit  sensor3 = P2^2; //Obstacle
sbit  sensor4 = P2^3; //Humidity
sbit  sensor5 = P2^4; //Smoke
sbit  DQ = P2^5;//Temperature

sbit  SCL=P2^6; //IIC Clock Pin
sbit  SDA=P2^7; //IIC Data Pin

uint  dis_data;
uchar Count=0;
uchar datSensor;
uchar BUF[8];
uchar buffer[64];
uchar temps[8];

void delay_nms(uchar k)
{
  uchar i,j;
  for(i=0;i<k;i++)
    for(j=0;j<255;j++){;}
}
void Delay5us()
{
  uchar k;
  for(k=0;k<16;k++){;}
}
void Delay5ms()
{
  uint k;
  for(k=0;k<560;k++){;}
}
void sDelay(int num)
{
  while(num--);
}
void Init_DS18B20(void)
{
  uchar x=0;
  DQ = 1;
  sDelay(8);
  DQ = 0;
  sDelay(80);
  DQ = 1;
  sDelay(14);
  x=DQ;
  sDelay(20);
}
uchar ReadOneChar(void)
{
  uchar i=0;
  uchar dat = 0;
  for (i=8;i>0;i--)
  {
   DQ = 0;
   dat>>=1;
   DQ = 1;
   if(DQ)
   dat|=0x80;
   sDelay(4);
  }
  return(dat);
}
void WriteOneChar(uchar dat)
{
  uchar i=0;
  for (i=8; i>0; i--)
  {
    DQ = 0;
    DQ = dat&0x01;
    sDelay(2);
    DQ = 1;
    dat>>=1;
  }
}
void ReadTemperature(void)
{
  uchar a=0;
  uchar b=0;
  uchar Data_L=0;

  Init_DS18B20();
  WriteOneChar(0xCC);
  WriteOneChar(0x44);
  Init_DS18B20();
  WriteOneChar(0xCC);
  WriteOneChar(0xBE);
  sDelay(500);
  a=ReadOneChar();
  b=ReadOneChar();
  Data_L=a&0X0F;

  temps[0] = (a/16+b*16)/10 +'0';
  temps[1] = (a/16+b*16)%10+'0';
  temps[2] = 0x2E;
  temps[3] = Data_L*10/16 +'0';
  temps[4] = (a/16+b*16)%10 +'0';
}
void BH1750_Start()
{
    SDA = 1;
    SCL = 1;
    Delay5us();
    SDA = 0;
    Delay5us();
    SCL = 0;
}
void BH1750_Stop()
{
    SDA = 0;
    SCL = 1;
    Delay5us();
    SDA = 1;
    Delay5us();
}
void BH1750_SendACK(bit ack)
{
    SDA = ack;
    SCL = 1;
    Delay5us();
    SCL = 0;
    Delay5us();
}
bit BH1750_RecvACK()
{
    SCL = 1;
    Delay5us();
    CY = SDA;
    SCL = 0;
    Delay5us();
    return CY;
}
void BH1750_SendByte(uchar dat)
{
    uchar i;
    for (i=0; i<8; i++)
    {
        dat <<= 1;
        SDA = CY;
        SCL = 1;
        Delay5us();
        SCL = 0;
        Delay5us();
    }
    BH1750_RecvACK();
}
uchar BH1750_RecvByte()
{
    uchar i;
    uchar dat = 0;
    SDA = 1;
    for (i=0; i<8; i++)
    {
        dat <<= 1;
        SCL = 1;
        Delay5us();
        dat |= SDA;
        SCL = 0;
        Delay5us();
    }
    return dat;
}

void                    Single_Write_BH1750(uchar
REG_Address)
{
    BH1750_Start();
    BH1750_SendByte(SlaveAddress);
    BH1750_SendByte(REG_Address);
    BH1750_Stop();
}
void Init_BH1750(void)
{
   Single_Write_BH1750(0x01);
}
void Multiple_Read_BH1750(void)
{
    uchar i;
    BH1750_Start();
    BH1750_SendByte(SlaveAddress+1);

    for (i=0; i<3; i++)
    {
        BUF[i] = BH1750_RecvByte();
        if (i == 3)
          BH1750_SendACK(1);
        else
          BH1750_SendACK(0);
    }
    BH1750_Stop();
    Delay5ms();
}
void conversion(uint temp_data)
{
    temps[0]=temp_data/10000+'0';
    temps[1]=(temp_data%10000)/1000+'0';
    temps[2]=(temp_data%1000)/100+'0';
    temps[3]=(temp_data%100)/10+'0';
    temps[4]=(temp_data%10)+'0';
}
float brightnessGet(){
     float tempv;
     Single_Write_BH1750(0x01);
     Single_Write_BH1750(0x10);
     delay_nms(90);
     Multiple_Read_BH1750();
     dis_data=BUF[0];
     dis_data=(dis_data<<8)+BUF[1];
     tempv=(float)dis_data/1.2;
            return tempv;
}
void Init232(void){
    SCON = 0x50;
    TMOD = 0x20;
    TH1 = 0xFD;    //9600bps@11.0592MHz
   TL1 = 0xFD;
   TR1 = 1;
}
void send_char_com(uchar ch)
{
    SBUF=ch;
    while (TI== 0);
    TI=0;
    ES=1;
}
void send_string_com(char *str)
{
   while(str && *str) {
    send_char_com(*str++);
  }
}
void main (void)
{
  float temp;
  Init232();
  Init_BH1750();
  ES  = 1;
  EA  = 1;

  while(1)
  {
    delay_nms(255);
    datSensor = 0;

    P2 = 0xFF;
    sensor1 = 0x1;
    sensor2 = 0x1;
    sensor3 = 0x1;
    sensor4 = 0x1;
    sensor5 = 0x1;
    datSensor = P2 & 0x1F;

    strcpy( buffer, "01&sud=");
    if(datSensor & 0x01) strcat( buffer, "0");
    else strcat( buffer, "1");

    strcat( buffer, "&hmn=");
    if(datSensor & 0x02) strcat( buffer, "1");
    else strcat( buffer,"0");

    strcat( buffer, "&obs=");
    if(datSensor & 0x04) strcat( buffer, "0");
    else strcat( buffer, "1");

    strcat( buffer, "&hmd=");
    if(datSensor & 0x08) strcat( buffer, "1");
    else strcat( buffer, "0");

    strcat( buffer, "&smk=");
    if(datSensor & 0x10) strcat( buffer, "0");
    else strcat( buffer, "1");

    ReadTemperature();
    strcat( buffer, "&tmp=");
    strcat( buffer, temps);

    temp=brightnessGet();
    conversion(temp);
    strcat(buffer, "&brt=");
    strcat(buffer, temps);
    strcat(buffer, "&lcn=CW215A");

    send_string_com(buffer);
  }
}
```

(2) Command Request and Execution

```
#include <REG52.H>
#include "string.h"

sbit LED  = P1^0;
sbit FAN  = P1^1;

unsigned int qmFlag = 0;
unsigned char onOff = '0';
unsigned char buffer[96];
unsigned char rec_cmd[10];


void Delaynms(unsigned int di){
unsigned int da,db;
  for(da=0;da<di;da++)
    for(db=0;db<100;db++);
}

void Init232(void){
   SCON =0x50;
   TMOD =0x20;
   TH1 =0xfA;
   IE |= 0x90;
   PCON = 0x80;
   TR1 =1;
   ET1 =0;
   ES=1;
   PS=1;
   EA =1;
}
void send_char_com( unsigned char ch)
{
    SBUF=ch;
    while (TI== 0);
    TI=0;
    ES=1;
}
void send_string_com(char *str)
{
   while(str && *str) {
    send_char_com(*str++);
  }
}
void main (void)
```

```
{
   Init232();
   ES  = 1;
   EA  = 1;
   ET0=1;
   TR0=1;
   LED = 0;  //off
   FAN = 0;  //off

 while(1)
 {
    if(onOff == '1')  LED = 0;   //off
    if(onOff == 'a')  LED = 1;   //on

    if(onOff == '7')  FAN = 0;   //off
    if(onOff == 'g')  FAN = 1;   //on

    if(onOff == '9')  {
            LED = 0;
            FAN = 0;
          } //off
    if(onOff == 'z')  {
            LED = 1;
            FAN = 1;
          } //on
   Delaynms(500);
    EA  = 0;
    send_string_com("lamp");
    EA  = 1;

    if(onOff == '1')  LED = 0;   //off
    if(onOff == 'a')  LED = 1;   //on
    if(onOff == '7')  FAN = 0;   //off
    if(onOff == 'g')  FAN = 1;   //on

    if(onOff == '9')  {
            LED = 0;
            FAN = 0;
          } //off
    if(onOff == 'z')  {
            LED = 1;
            FAN = 1;
          } //on
    Delaynms(500);
    EA  = 0;
    send_string_com("fan");
    EA  = 1;
  }
}
void ser_int (void) interrupt 4
{
   unsigned char UART_buff;
   if(RI == 1) {
     RI = 0;
     UART_buff = SBUF;
     if(UART_buff == '?')
          qmFlag = 1;
     if((qmFlag == 1)&&(UART_buff != '?')){
          onOff = UART_buff;
          qmFlag = 0;
     }
   }
   else
      TI = 0;
}
```

2. Working Mode Setting of WiFi Transceiver RN-370M

(1) For Context Update

```
$$$; // Turn to command mode
Scan; // Scan available wireless network
Set wlan ssid the-network-to-join; // Connect to the objective network
Save; //Save configuration
Reboot; //Restart
// set configuration
Set ip proto 18; // Turn HTTP mode=0x10 and TCP mode=0x02
Set ip host 10.0.0.11;//Web server DNS
Set ip remote 80; //Set web server port
Set sys autoconn 1; //Automatically connect when ready
Set com remote GET$/smarthome/contextUpdate.php?ID=; //Sample server application
Set uart mode 2; //Automatically connect using data trigger mode
Save; //Save the configuration to the configuration file
Reboot; //Reboot so that the settings take effect
```

(2) For Service Request

```
$$$;
Scan;
Set wlan ssid the-network-to-join;
Save;
Reboot;
Set ip proto 18;
Set ip host 10.0.0.11;
Set ip remote 80;
Set sys autoconn 1;
Set com remote GET$/smarthome/deviceControl.php?command=;
Set uart mode 2;
Save;
Reboot;
```

3. Service Applications for Context Update and Context Request in PHP

(1) Context Update

```
<?php
    require_once "./DBService.php";
    $data= array();
    $dbcon = new DBService();
    $t_today = date("dmY");
    $t_tofday = date("His");
    $datetime = $t_today.'-'.$t_tofday;
    $c_brightness = $_GET["brt"];
    for(;;){
      if(($c_brightness[0] == '0')&&(strlen($c_brightness)>1))
         $c_brightness = substr($c_brightness,1);
      else break;
    }
    if(intval($c_brightness <150))
    $c_brightness='L';
    else if(intval($c_brightness >300))
    $c_brightness='H';
    else $c_brightness='M';
    $result = $dbcon->contextUpdate($_GET["sud"],$c_brightness,$_GET["hmn"],$_GET["obs"],$_GET["hmd"],$_GET["smk"],$_GET["tmp"],'CW215A',$_GET["ID"],$datetime);
    if($result == '1')
      echo "OK";
    else
      echo "ERR";
?>
```

(2) Context Request

```
<?php
require_once "./DBService.php";
$id = $_GET["ID"];

$data= array();
$dbcon = new DBService();

$dom = new DOMDocument("1.0","UTF-8");
header("Content-Type: text/xml");

$root = $dom->createElement("context");
$dom->appendChild($root);

$s = $dbcon->contextRequest($id);
while($data = mysql_fetch_array($s)){

    $item = $dom->createElement("contextItems");
    $root->appendChild($item);

    $bn = $dom->createElement("s_brightness");
    $item->appendChild($bn);
    $bns = $dom->createTextNode($data['s_brightness']);
    $bn->appendChild($bns);

    $hm = $dom->createElement("s_human");
    $item->appendChild($hm);
    $hum = $dom->createTextNode($data['s_human']);
    $hm->appendChild($hum);

    $ob  = $dom->createElement("s_obstacle");
    $item->appendChild($ob);
    $obs = $dom->createTextNode($data['s_obstacle']);
    $ob->appendChild($obs);

    $hd  = $dom->createElement("s_humidity");
    $item->appendChild($hd);
    $hdt = $dom->createTextNode($data['s_humidity']);
    $hd->appendChild($hdt);

    $sm = $dom->createElement("s_smoke");
    $item->appendChild($sm);
    $smk = $dom->createTextNode($data['s_smoke']);
    $sm->appendChild($smk);

    $tm = $dom->createElement("s_temperature");
    $item->appendChild($tm);
    $tmp = $dom->createTextNode($data['s_temperature']);
    $tm->appendChild($tmp);

    $nd = $dom->createElement("w_node");
    $item->appendChild($nd);
    $nde = $dom->createTextNode($data['w_node']);
    $nd->appendChild($nde);

    $ti = $dom->createElement("c_time");
    $item->appendChild($ti);
    $tim = $dom->createTextNode($data['c_time']);
    $ti->appendChild($tim);

}
echo $dom->saveXML();
?>
```

4. Rule Matching Algorithm for Service Customisation in PHP

```
<?php
   require_once "./DBService.php";
   require_once "./Itemdepth.php";
   require_once "./Itemrange.php";
   $ruleKeys = array(0 =>'s_brightness',1 =>
's_human',2 => 's_weather',3 =>
's_temperature',4 => 's_noise,5 => 's_humidity',
6 =>'s_smoke',7 =>'c_time',8 =>'c_action', 9
=>'u_command');
   $c_array = array();
   $r_array = array();
   $ii = 0;
   $min_dist = 1;
   $ruleLength = count($ruleKeys)-2;
   $operation = 'Error Operation!';
   $theCommand = $_GET["command"];
   $dbcon = new DBService();
   $itemdepth = new Depthofitem();
   $itemrange = new Rangeofitem();
   $c = $dbcon->contextRequest();
   while($context = mysql_fetch_array($c))
   {
      for($i = 0;$i < $ruleLength;$i ++)
        $c_array[$i] = $context[$ruleKeys[$i]];
   }
   $r = $dbcon->ruleRequest($theCommand);

   while($rules = mysql_fetch_array($r))
   {
     $r_len = 0;
     $dist = 0;
     for($i = 0;$i < $ruleLength;$i ++)
        $r_array[$i] = $rules[$ruleKeys[$i]];
     for($i = 0;$i < $ruleLength;$i ++){
        if($r_array[$i]!='?') {
             if(is_numeric($r_array[$i])==false)
                  $dist += (1 - 2*$itemdepth-
>LCA($c_array[$i],$r_array[$i])/($itemdepth-
>Depth($c_array[$i])+$itemdepth-
>Depth($r_array[$i])));
             else
                  $dist += abs($c_array[$i]-
$r_array[$i])/$itemrange-
>Itemrange($ruleKeys[$i]);
             $r_len ++;
        }
     }
     $ii ++;
     $dist /= $r_len;
     if($dist < $min_dist){
        $min_dist = $dist;
        if($rules[$ruleKeys[$ruleLength-1]] != '?')
            $operation =
$rules[$ruleKeys[$ruleLength-1]];
        else
            $operation =
$rules[$ruleKeys[$ruleLength-2]];
     }
   }
   echo '?'.$operation;
?>
```

5. Rule Generation Algorithm in PHP

```
<?php
require_once "./DBService.php";
class Utilities{
  function ruleGeneration($theCommand){
   $ruleKeys  =  array(0  =>'s_brightness',1  =>
's_human',2        =>        's_weather',3        =>
's_temperature',4       =>       's_sound',5       =>
's_humidity', 6 =>'s_smoke',7 =>'timeofday');
   $c_array = array();
   $d_array = array();
   $dm = array();
   $red ='0' ;
   $id_max = 0;
   $nattributes = count($ruleKeys);
   $nsamples = 20;
   $cstring='0';
   $dstring='0';
   $dbcon = new DBService();
   $c = $dbcon->contextRequest();
   while($context = mysql_fetch_array($c))
   {
     $id_max = $context['ID'];
   }
   for($j=$id_max;$j>($id_max-$nsamples);$j--
){
      $cstring='0';
      $dstring='0';
      $c = $dbcon->contextRequestwithid($j);
      while($context = mysql_fetch_array($c))
      {
        for($i=0;$i<$nattributes;$i++)
          $cstring.=$context[$ruleKeys[$i]];
      }
      $d = $dbcon->decisionRequestwithid($j);
      while($decision = mysql_fetch_array($d))
      {
        $dstring.= $decision[$theCommand];
      }
      $c_array[$id_max-$j]= substr($cstring,1);
      $d_array[$id_max-$j] = substr($dstring,1);
   }
   //Create the Discernibility Matrix
   for($i=0;$i<$nsamples;$i++){
```

```
for($j=0;$j<$nsamples;$j++)$dm[$i][$j]='0';
        for($j=$i+1;$j<$nsamples;$j++){
         $rstring = '0';
         if($d_array[$i]== $d_array[$j]) {
             $rstring = '00';
             }else{
                 for($k=0;$k<$nattributes;$k++) {
                   $cstringi = $c_array[$i];
                            $cstringj = $c_array[$j];
                            if($cstringi[$k]                   !=
$cstringj[$k])
                            $rstring.= $k+1;
                          }
             }
             $dm[$i][$j] = substr($rstring,1);
        }
    }
    // Attribute Reduction
    for(;;){
      $stemp ='0';
      $flag1 = '0';
      $flag2 = '0';
      for($i=0;$i<$nsamples;$i++){
         for($j=0;$j<$nsamples;$j++){
                $stemp = $dm[$i][$j];
                if($stemp != '0') {
                   $flag2 = '1';
                   if((strlen($stemp)== 1)){
                   $red.= $stemp;
                            $flag1 = '1';
                   }
                }
             }
      }
      if($flag1 == '1') {
        for($i=0;$i<$nsamples;$i++)
          for($j=0;$j<$nsamples;$j++){
                 for($k=0;$k<strlen($red)-1;$k++) {

if(strpos($dm[$i][$j],$red[$k+1]) !== false)
                            $dm[$i][$j]='0';
                        }
              }
         }
         else if($flag2 == '1'){
           $c_min = $nattributes;
                  $c_num = 0.0;
               $m_array=array();
               for($k=0;$k<$nattributes;$k++)
                   $m_array[$k] = 0.0;
               for($k=0;$k<$nattributes;$k++){
                  for($i=0;$i<$nsamples;$i++)
               for($j=0;$j<$nsamples;$j++){

if(strpos($dm[$i][$j],strval($k+1)) !== false)
                          $m_array[$k]                   +=
1/strlen($dm[$i][$j]);
                   }
               }
               for($k=0;$k<$nattributes;$k++){
                  if($m_array[$k] > $c_num)
                       $c_num = $m_array[$k];
               }
               for($k=0;$k<$nattributes;$k++){
                   if($m_array[$k] == $c_num) break;
               }
               $red.=strval($k+1);
               for($i=0;$i<$nsamples;$i++)
            for($j=0;$j<$nsamples;$j++){
                        if
(strpos($dm[$i][$j],strval($k+1)) !== false)
                       $dm[$i][$j] = '0';
                 }
               }
         if(($flag1 == '0')&&($flag2 == '0')) break;
       }
       $red = substr($red,1);
               return $red;
    }
}
?>
```

6. Testing Results of Case Study 1

(1) Operation Time of the Touch Screen Manual Input Method (Seconds)

	iPhone4S	Nexus S	HD2	N8
1	18.531	18.843	19.262	23.139
2	18.165	17.673	22.879	24.841
3	19.779	18.261	20.656	23.173
4	18.571	18.787	21.076	22.674
5	18.08	17.2	19.913	23.505
6	23.746	16.598	19.932	27.907
7	20.019	18.861	20.494	22.471
8	20.875	18.891	20.12	22.238
9	21.96	17.166	21.682	20.397
10	18.556	16.882	20.349	19.659

(2) Operation Time of the Barcode Scanning Method (Seconds)

	iPhone4S	Nexus S	HD2	N8
1	2.451	8.36	5.261	14.78
2	2.583	8.664	8.047	12.463
3	2.452	12.77	9.547	12.983
4	2.243	9.375	3.405	11.465
5	3.054	7.574	19.696	18.94
6	3.05	6.371	14.262	11.852
7	2.705	5.283	3.337	16.818
8	2.6	6.717	5.143	11.294
9	2.287	9.438	11.63	10.839
10	2.582	6.474	17.891	11.399

11	19.027	16.686	19.069	19.266
12	19.281	16.35	23.27	21.176
13	19.085	18.28	19.692	19.043
14	17.873	18.185	19.604	21.113
15	21.145	18.776	20.193	22.823
16	18.784	19.587	19.411	24.853
17	18.77	17.854	19.449	21.891
18	18.189	16.564	20.652	20.607
19	19.411	19.579	19.793	20.014
20	18.152	17.894	20.378	20.915

11	2.885	14.088	11.963	10.767
12	3.958	9.154	7.339	10.459
13	2.161	6.369	28.704	10.427
14	3.105	9.053	17.527	16.316
15	2.928	5.369	4.379	16.895
16	3.253	17.477	14.494	17.101
17	2.76	5.916	19.866	11.241
18	2.312	5.854	16.452	11.555
19	2.408	4.95	9.624	11.177
20	2.758	5.51	16.379	11.737

7. Testing Results of Case Study 2

(1) Running Time of Rule Generation Algorithm on Two Testing Platforms

Number of Rules	iMac Local Host Server (Seconds)	Remote Testing Server (Seconds)
5	6.9141387939453E-6	1. 0013580322266E-5
10	0.05935001373291	0.0021438598632812
15	0.097636938095093	0.0035839080810547
20	0.14931583404541	0.0042879581451416
25	0.21001696586609	0.0055959224700928
30	0.28509998321533	0.0072400569915771
35	0.31743812561035	0.0088109970092773
40	0.37803816795349	0.010339021682739
45	0.4365770816803	0.011605024337769
50	0.45534294128418	0.013478994369507
55	0.48700904846191	0.013916015625
60	0.58310103416443	0.014717102050781
65	0.65155005455017	0.017521142959595
70	0.70108318328857	0.017922878265381
75	0.75324821472168	0.019410848617554
80	0.76536107063293	0.020502090454102
85	0.88367199897766	0.021950960159302
90	0.9376118183136	0.023576974868774
95	0.98812484741211	0.024829864501953
100	1.0763549804688	0.026623964309692

(2) Running Time of Rule Generation Algorithm on iMac Local Host Server

Number of Context Items	Data Preparing Time (Seconds)	Algorithm Time (Seconds)	Total Time (Seconds)
5	0.11827707290649	0.00029110908508301	0.11856818199158
10	0.23557114601135	0.0010788440704346	0.23664999008179
15	0.35156011581421	0.0041718482971191	0.35573196411133
20	0.49317193031311	0.0018751621246338	0.49504709243774
25	0.58999800682068	0.003521203994751	0.59351921081543
30	0.72665810585022	0.005824089050293	0.73248219490051
35	0.84392905235291	0.0094609260559082	0.85338997840881

40	0.96281409263611	0.01959490776062	0.98240900039673
45	1.0781519412994	0.051945924758911	1.1300978660583
50	1.1952328681946	0.11716413497925	1.3123970031738
55	1.3303608894348	0.22973823547363	1.5600991249084
60	1.4306700229645	0.39297986030579	1.8236498832703
65	1.5448460578918	0.6586709022522	2.203516960144
70	1.6642320156097	0.95733189582825	2.621563911438
75	1.7491409778595	1.4224591255188	3.1716001033783
80	1.8769130706787	1.9836709499359	3.8605840206146
85	1.9639658927917	2.7159020900726	4.6798679828644
90	2.0930709838867	3.643424987793	5.7364959716797
95	2.2312459945679	4.7367050647736	6.9679510593414
100	2.3448841571808	6.64138007164	8.9862642288208

(3) Running Time of Rule Generation Algorithm on Remote Testing Server

Number of Context Items	Data Preparing Time (Seconds)	Algorithm Time (Seconds)	Total Time (Seconds)
5	0.0039792060852051	4.7922134399414E-5	0.0040271282196045
10	0.0054080486297607	0.00012588500976562	0.0055339336395264
15	0.0090339183807373	0.00025296211242676	0.0092868804931641
20	0.011054039001465	0.00043106079101562	0.01148509979248
25	0.013361930847168	0.0006561279296875	0.014018058776855
30	0.016549110412598	0.00091695785522461	0.017466068267822
35	0.018509149551392	0.0012688636779785	0.01977801322937
40	0.021145105361938	0.0016078948974609	0.022753000259399
45	0.024431943893433	0.002018928527832	0.026450872421265
50	0.027137041091919	0.002479076385498	0.029616117477417
55	0.028044939041138	0.0029749870300293	0.031019926071167
60	0.031302928924561	0.0035340785980225	0.034837007522583
65	0.034090042114258	0.004188060760498	0.038278102874756
70	0.037007808685303	0.0048561096191406	0.041863918304443
75	0.040647983551025	0.0054740905761719	0.046122074127197
80	0.04284405708313	0.0062220096588135	0.049066066741943
85	0.04432201385498	0.0070838928222656	0.051405906677246
90	0.049053907394409	0.0079469680786133	0.057000875473022
95	0.051819801330566	0.0088300704956055	0.060649871826172
100	0.056939125061035	0.0098040103912354	0.066743135452271