Distributed Support for Intelligent Environments

A thesis submitted for the degree of
Doctor of Philosophy at The Australian National University

Teddy Mantoro

Department of Computer Science
The Australian National University
ACT 0200, Australia

24 April 2006

Declaration

I declare that the research described in this thesis is my own original work during my PhD study under the supervision of the members of advisory panel, i.e. Assoc. Prof. Christopher W. Johnson (chair and main supervisor), Assoc. Prof. Bob Kummerfeld (co-supervisor) and Dr. Ken Taylor (co-supervisor), except where otherwise acknowledged in the text.

Teddy Mantoro
April 2006

Publications

Journals:

1. Mantoro, T., and C. W. Johnson, **Fusing Sensors to Enabling Intelligent Responses in an Active Office**, Submitted to the Journal of Pervasive and Mobile Computing (PMC) by Elsevier, ISSN 1574-1192, December 2004.
2. Mantoro, T., and C. W. Johnson, **Instance-Based Learning Methods for the Best Estimation of Topological User Location in Pervasive Environments**, Submitted to the International Journal of Mobile Computing and Communication Review (MC^{2}R), ACM SIGMOBILE , May 2005.
3. Mantoro, T., and C. W. Johnson, **Location Based User Activity in a Pervasive Computing Environment**, Submitted to the International Journal of Pervasive Computing and Communication, ISSN (Online): 1742-738X - ISSN (Paper): 1742-7371, June 2005.

Conferences:

1. Mantoro, T., and C. W. Johnson, "**Design Space: Enabling 'Unregistered User' to Access His Own Content.**" The Seventh International Conference on Ubiquitous Computing (UbiComp'05) Workshop 6 - The Spaces in-between: Seamful vs. Seamless Interactions, Tokyo, Japan, 11-14 September 2005.
2. Mantoro, T., and C. W. Johnson. ***ηk*-Nearest Neighbour algorithm for Estimation of Symbolic User Location in Pervasive Computing Environments.** Accepted to the IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks (WoWMoM), Taormina, Italy, 13-16 June 2005
3. Mantoro, T., "**Understanding User Activity in Distributed Intelligent Environments**", Proceeding of the Third IEEE Conference on Computing and Intelligent System (Kommit'04), ISSN-1411-6286, Jakarta, Indonesia, 14-25 August 2004.
4. Mantoro, T., and C. W. Johnson, **DiCPA: Distributed Context Processing Architecture for an Intelligent Environment**, Proceeding of the Western

Multiconference (WMC): Communication Networks And Distributed Systems Modeling And Simulation Conference (CNDS'04), San Diego, California, 19-22 January 2004.

5. Mantoro, T., and C. W. Johnson, **User Mobility Model in an Active Office**, LNCS 2875, Proceeding of the European Symposium on Ambient Intelligence (EUSAI'03), Eindhoven, The Netherlands, 3-4 November 2003.
6. Mantoro, T. **User Location and Mobility for Distributed Intelligent Environment**, Adjunct Proceedings, The Fifth International Conference on Ubiquitous Computing (UbiComp'03), Seattle, Washington, USA, 12-15 October 2003.
7. Mantoro, T., and C. W. Johnson**, "Location History in a Low-cost Context Awareness Environment",** Workshop on Wearable, Invisible, Context-Aware, Ambient, Pervasive and Ubiquitous Computing, Australian Computer Science Communications, Volume 25, Number 6, Adelaide, Australia, February 2003.

Acknowledgement

The ANU department of Computer Science has provided me with a great atmosphere for my PhD research for the past three years. It has been a great privilege to be surrounded by so many excellent computer scientists, both theoretical and experimental, particularly under the supervision of Assoc. Prof. Chris Johnson who brought me to the research and community of Smart Internet Technology. Discussion with him has been a time of great privilege for me and seemingly endless enthusiasm and imagination generated new perspectives: his honest critiques came in the form of deep and inevitably challenging questioning. He is my mentor whose approach is defined by the words of Glaser (1995), who said: "*Grab one corner of the problem and go! Start doing it*". For such a rich and overwhelming introduction to the world of research I am deeply grateful. I am also grateful for my advisory panel who gave me a lot of invaluable feedback, i.e., Assoc. Prof. Bob Kummerfeld from the University of Sydney and Dr. Ken Taylor from the Commonwealth Scientific and Industrial Research Organisation (CSIRO).

I would like to thanks Prof. Matthew James, as the Head of the Department of Engineering (2001-2002), who supported me at the beginning of my study by allowing me to continue working part time as a computer system administrator in the Department while I pursued my studies and for providing me with a fee-waiver scholarship from March 2002-Oct 2002; Prof. Michael Cardew-Hall, as the Head of the Department of Engineering (2002-present) who continuously supported me during my studies and Rob Gresham my supervisor in the Department of Engineering who is very supportive and understanding, he did everything he could to make sure it would be easier for me to do my job.

The community in which I have worked throughout this PhD has been extremely generous, both with knowledge and money, i.e.,

Smart Internet Technology – CRC (http://www.smartinternet.com.au), especially Prof. Darrell Williamson as CEO of SIT-CRC, has provided me with financial support for my PhD from January 2003 to June 2005 (2.5 years) and give me the opportunity to participate in a series of SIT-CRC conferences and to publish my work.

Faculty of Engineering and Information Technology (FEIT) – ANU, through the Faculty Grant Research Scheme (FRGS), that provided me with a one year grant for research support (January-December 2004)

ANU National Institute of Engineering and Information Sciences (NIEIS) through its NIEIS travel award has provided me with travel support for a conference (January 2004).

Department of Computer Science, especially Assoc. Prof. Chris Johnson as its Head of Department, and Department of Engineering, both have provided me with varying financial support, especially travel support to conferences.

I also would like to thank several people especially Dr. Eric McCreath and Prof. John Lloyd for valuable discussions in the area of machine learning, several SIT-CRC PhD scholars especially Adam Hudson, Dan Cutting, David Carmichael, Mark Assad, Michael Avery and Derek Corbett from the University of Sydney, with whom I shared experiences when visiting Media Lab–MIT, Max Planck Lab, Saarland University, and

DFKI Lab in Germany, as well as the DTG lab, Cambridge University UK (October-November 2003).

Very special thanks to Andrew Wilkinson who become my partner in discussing technical experiments with several fixed and precise sensors and Kanwar Sidhu, both my colleagues as Computer System Administrators in the Department of Engineering along with my PhD study.

I would like to express my very special thanks to several of my very good friends who voluntarily helped me with English grammar, to John Shelton and Albert Deme for their early help of publishing conference papers, to Else Sugito for my early thesis draft and conferences paper, to Tony Flynn for his invaluable effort in making my thesis more readable, especially with English grammar and usage, punctuation, voice and tone and to Dr. Michelle McCann during my thesis revision.

Finally, I would like to thank my real live team: Media, Mamo and Yutta who sacrificed countless weekend hours for me during this PhD. This work is dedicated to them.

Glossary of Abbreviations

AmI	Ambient Intelligence
AI	Artificial Intelligent
ANN	Artificial Neural Network
ANU	The Australian National University
AP	Access Point
API	Application Program Interface
ASR	Automatic Speech Recognition
Aura	An Architectural Framework for User Mobility in Ubiquitous Computing Environments. Carnegie Mellon University: "Distraction-free Ubiquitous Computing"
BDA	Bluetooth Device Address
Bluetooth	Short distance wireless cable replacement technology
Bluejacking	The sending of unsolicited message over Bluetooth to Bluetooth-enable devices, such as mobile phones, PDAs Smart Phones or Laptops
BT	Bluetooth
CAIP	Centre for Advanced Information Processing
CIPE	Crypto Internet Protocol Encapsulation
CLIPS	C Language Integrated Production System
CMU-TMI	Carnegie Mellon University – Triangulation Mapping Interpolation
CS	Computer Science
CSIT	Computer Science and Information Technology
DB	Database
DCS	Department of Computer Science
DHCP	Dynamic Host Configuration Protocol
DHT	Distributed Hash Table
DiCPA	Distributed Context Processing Architecture
DNS	Domain Name Server
DSTO	Defence Science and Technology Organisation
ECIS	European Conference on Information Systems
ECSE	Experimental Computer Science and Engineering
Ekahau	Commercial software which has capability to locate location in wireless (IEEE 802.11) local area network environment.
ESPRIT MUSiC	European information technologies (IT) programme (ESPRIT) Measurement of Usability in Context
FEIT	Faculty of Engineering and Information Technology
GPRS	General Package Radio Service
GPS	Global Position System
GSM	Global System for Mobile Communications
HCI	Human Computer Interaction
HTTP	Hypertext Transfer Protocol
ICMP	Internet Control Message Protocol
ID	Identification
IE	Intelligent Environment

IEC	International Electrotechnical Commission
IEEE	Institute of Electrical and Electronics Engineers
IMAP	Internet Message Access Protocol
IP	Internet Protocol
IrDA	Infra-red Data Association
IROS	Interactive Room (iRoom) Operating System
ISO	International Standardization Organization
JESS	Java Expert System Shell
JINI	An open software architecture that enables Java Dynamic Networking for building distributed systems that are highly adaptive to change.
JSAPI	Java Speech Application Program Interface
JXTA	Stands for Project Juxtapose (more simply, JXTA)
k-NN	*k*-Nearest Neighbour
LCE	Laboratory for Communications Engineering, Cambridge
LDAP	Lightweight Directory Access Protocol
MAC address	Media Access Control address
MCDM	Multiple Criteria Decision Making
MCRDR	Multiple Classification Ripple Down Rules
MIT	Massachusetts Institute of Technology
MRTG	Multi Router Traffic Grapher
NAPTR	Naming Authority Pointer
NAT	Network Address Translation
Nibble	A Wi-Fi location service that uses Bayesian networks to infer the location of a device.
NIS(YP)	Network Information Service (Yellow Page)
ntop	Network TOP – A network traffic probe that shows the network usage
P2P	Peer-to-Peer
PAN	Personal Area Network
PANU	Personal Access Network User
PARC	Palo Alto Research Center (Xerox)
PCA	Principal Component Analysis
PDA	Personal Digital Assistance
POP3	The PPTP server solution for Linux
PoPToP	The PPTP server solution for Linux
PPTP	Point to Point Tunnelling Protocol
PPPd	Point-to-Point Protocol daemon
PURL	Persistent Unique Resolution Protocol
RADAR	A radio-frequency (RF) based system for locating and tracking users inside buildings.
RBAC	Role-Based Access Control
RFC	Request For Comment
ROADMAP	Role-Oriented Analysis and Design for Multi-Agent Programming; a generic meta-model for describing multi-agent systems

RPC	Remote Procedure Call
RFID	Radio Frequency Identification
RJ45	Registered Jack - Type 45
RPC	Remote Procedure Call
RR	Resources Record
SADT	Structured Analysis and Design Technique
SEA	Smart Environment Agent
SNMP	Simple Network Management Protocol
SOM	Self Organising Map
SPA	Smart Personal Assistant
SpeechCA	Speech Context-Aware
SQL	Select Query Language
UDP	User Datagram Protocol
UMTS	Universal Mobile Telecommunications System
UPnP	Universal Plug and Play
URI	Unique Resolution Identifier
URL	Unique Resolution Locator
URN	Unique Resolution Name
USB	Universal Serial Bus
UWB	Ultra-Wideband
VNC	Virtual Network Computing
VPN	Virtual Private Network
WAAS	Wide Area Augmentation System
WiFi	Wireless Fidelity, the Alliance to certify interoperability of IEEE 802.11
WiMedia	Brand for high data-rate, wireless multimedia networking applications operating in a WPAN
WLAN	Wireless Local Area Network
WPAN	Wireless Personal Area Network
WVLAN	Wireless Virtual Local Area Network
XDM	X Display Manager (a graphical windows which manage remote X servers and provide login prompts to remote 'X terminals' or to manage the users X session)
X-terminals	A machine with a network connection, keyboard, mouse and monitor, configured to run the X Windows System to connect to an application server on the network
Zigbee	A combination of HomeRF Lite and the IEEE 802.15.4 specification
ηk-NN	ηk-Nearest Neighbour

Abstract

This thesis describes research on methods for Ubiquitous/Pervasive Computing to better suit users in an Intelligent Environment. The approach is to create and equip a computing environment, such as our Active Office, with technologies that can identify user needs and meet these need in a timely, efficient and unobtrusive manner.

The critical issues in the Intelligent Environment are how to enable transparent, distributed computing to allow continued operation across changing circumstances and how to exploit the changing environment so that it is aware of the context of user location, the collection of nearby people and objects, accessible devices and changes to those objects over time.

Since the Intelligent Environment is an environment with rapid and rich computing processing, the distributed context processing architecture (DiCPA) was developed to manage and respond to rapidly changing aggregation of sensor data. This architecture is a scalable distributed context processing architecture that provides: 1. continued operation across changing circumstances for users, 2. the collection of nearby people and objects, 3. accessible devices and 4. the changes to those objects over time in the environment. The DiCPA approach focuses on how the Intelligent Environment provides context information for user location, user mobility and the user activity model. Users are assumed mobile within the Intelligent Environment and can rapidly change their access to relevant information and the availability of communications and computational resources.

Context-Aware Computing is a new approach in software engineering for Intelligent Environment. It is an approach in the design and construction of a context-aware application that exploits rapid changes in access to relevant information and the availability of communication and computing resources in the mobile computing environment. The goal of Context-Aware Computing is to make user interaction with the computer easier in the smart environment where technology is spread throughout (pervasive), computers are everywhere at the same time (ubiquitous) and technology is embedded (ambient) in the environment. Context-aware applications need not be difficult, tedious or require the acquisition of new skills on the part of the user. They should be safe, easy, simple to use and should enable new functionality without the need to learn new technology. They should provide relevant information and a simple way for a user to manage.

The Intelligent Environment requires a context-aware application to improve its efficiency and to increase productivity and enjoyment for the user. The context awareness mechanism has four fundamental cores i.e. identity (who), activity (what), location (where) and timestamp (when). Based on DiCPA architecture, the model of user location (where), user mobility (where), user activity (what) and Intelligent Environment response (what) were developed. Prototypes were also developed to proof the Context-Aware Computing concept in the Intelligent Environment.

An Intelligent Environment uses the multi-disciplinary area of Context-Aware Computing, which combines technology, computer systems, models and reasoning, social aspects, and user support. A “good quality” project for Context-Aware Computing requires *core content* and provides *iterative evaluation* processes, which has two types of iteration: design and product iteration of the evaluation. The aim of the development of an evaluation program in Context-Aware Computing is to determine what to test, how to test and the appropriate metrics to use. This work presents the metrics for a good quality project in the Context-Aware Computing area, which is followed by the evaluation of the prototypes of this work.

Table of Contents

List of Figures

List of Tables

Chapter 1

INTRODUCTION

1.1 General Description of an Intelligent Environment

In the past decade the computer environment has changed over time, with user situations in a computing environment also becoming more complicated. The changes have affected the user profile, going from a single profile for a single device to many profiles and many devices for many users, as well as in the configuration of the network. One of the impacts on the enabling technology sees movement from a wired network to a wireless network, from a Local Area Network to a Personal Area Network or to a Wide Area Network.

The big issue in the Intelligent Environment is how to enable transparent distributed computing to allow continued operation in a seamless manner across a changing circumstance: how to exploit the changing environment so that it is aware of the context of user location, the collection of nearby people and objects, accessible devices and changes to those objects over time. Distributed computing in this context is the computer environment where data and processing are distributed among several computing systems (nodes) but control, such as network control, process coordination, synchronisation and scheduling, is centralised (Berson 1996: p64).

Scientists in the Intelligent Environment area are researching ways to make embedded and Ubiquitous Computing work better for users by creating and equipping a computing environment, such as an active home or an active office, with technologies that can identify user needs and meet them speedily, efficiently and unobtrusively.

One of the new approaches in software engineering is Context-Aware Computing. Context-Aware Computing is an approach in the design and construction of a context-aware application that exploits rapid changes in access to relevant information and the availability of communication and computing resources in the mobile computing environment. The goal of Context-Aware Computing is to make user interaction with the computer easier in the smart environment where technology is spread throughout (pervasive), computers are everywhere at the same time (ubiquitous) and technology is embedded (ambient) in the environment. It need not be difficult, tedious or require the acquisition of new skills on the part of the user. It should be safe, easy, simple, and should enable new functionality without the need to learn a new technology. It should provide relevant information and a simple way for a user to deal with it.

A context-aware application in the ubiquitous environment requires the location of a person within an environment (Jiang and Steenkiste 2002, Schilit et al. 1994). This can be obtained using wireless connections in devices that are normally carried for other purposes, for example, a mobile phone, PDA or smart phone. The location of these devices, and hence the person with them, is determined by a mixture of precise, proximate and predicted location sensors. The data from these sensors is turned into a predictor to precisely locate the device, and thus the person. Once a user is located, services and responses can be delivered based on the current situation.

An Intelligent Environment is an environment with rapid and rich computing processing using distributed context processing architecture. We have designed DiCPA as an example architecture to manage and respond to the rapidly changing aggregation of sensor data. This architecture is a scalable distributed context processing architecture that provides continued operation across changing circumstances for users, the collection of nearby people and objects, accessible devices and the changes to those objects over time in the environment. This approach focuses on how the Intelligent Environment provides context information for user location, user mobility and the user activity model. The context information of user activity can be used to characterise the user situation which enables the smart environment to respond to the user. With users moving around the Intelligent Environment, they can rapidly change their access to relevant information, and the availability of communications and computational resources.

There are four important variables in a context awareness mechanism i.e. identity (who), activity (what), location (where) and timestamp (when). Location is the most crucial aspect of context awareness for mobile users (Jiang and Steenkiste 2002, Schilit et al. 1994), such as finding nearest resources, navigation, locating objects and people. Location awareness will be discussed in detail in Chapters 4 and 5.

In a context-awareness mechanism, the scope of the 'who' question could be user identity, persona, profile, personalisation and user model. The scope of the 'what' question is the user's activity. These 'who' and 'what' questions are the most complex part in the Intelligent Environment, and they can be represented by user mobility, for example by finding patterns in the user's changing locations. Thus, a context awareness mechanism, based on user's locations and time stamps, could lead to a user's activity. User Mobility is discussed in Chapter 6, followed by the discussion of User Activity in Chapter 7.

These awareness-mechanisms bring the computer into the user's daily activity. This thesis explores computer capabilities in the recognition of user location, user activity and social context, as defined by the presence of other people, and the assistance of people with a variety of activities at work.

A model of an Intelligent Environment can be an Active Office, Active Home, Active Store or Active Supermarket. In this work, an Active Office is introduced as an implementation model of an Intelligent Environment. An Active office is defined as a normal office, which consists of several normal rooms with minimal additional decoration (minimal intrusive detectors and sensors), and with no requirement to attach devices (badging) on the user. An Active Office uses wireless communication, i.e. Bluetooth and WiFi, to enable user mobility. Wireless sensors that sense based on proximate visibility and received signal strength and signal quality in these wireless systems (PAN – Bluetooth - and WLAN - 802.11b,b+,g) were investigated to detect users in an Active Office.

Our Active Office uses a scalable distribution context processing architecture (DiCPA) to manage and respond to rapidly changing aggregation of sensor data. This architecture is based on the Merino service layers architecture (Kummerfeld, Quigley et al. 2003). The Merino architecture has four tightly coupled layers to manage the interaction between users and the environment in the Active Office, by detecting the user's change of location and transformation of the raw sensors data through application programs to deliver service while the user is on the move. The infrastructure supports interoperability of context-sensors/widgets and applications on heterogeneous platforms (Dey, Abowd et al. 1999; Kummerfeld, Quigley et al. 2003).

In order for an Active Office to provide services to users, it must be able to detect its current state/context and determine what responses to take based on that context. This thesis shows 'proof of concept' and 'proof of performance' of an Intelligent Environment by developing an architecture with a scalable distribution context processing capability to manage and respond to rapidly changing aggregation of sensor data in Chapter 3, and showing how the environment gives responses in Chapter 8.

Intelligent Environments use the multi-disciplinary area of Context-Aware Computing, which combines technology, computer systems, models and reasoning, social aspects, and user support. A "good quality" project for Context-Aware Computing requires *core content* (Section 9.2) and provides *iterative evaluation* processes (Section 9.7), which has two types of iteration: design and product iteration of the evaluation. The aim of the development of an evaluation program in Context-Aware Computing is to determine what to test, how to test and the appropriate metrics to use. Section 9.5 presents the metrics for a good quality project in Context-Aware Computing area and Section 9.10 appraises the evaluation of this work.

1.2 Problem Definition

Dealing with an Intelligent Environment where the computer environment changes with time and user situations become more complicated presents two sets of problems. Firstly, the interaction of the user with the changing environment must be considered. Secondly, the design of response to the change of user situation in the dynamic environment capability must be considered.

As Intelligent Environment has the capability to detect, respond and assist users with a variety of activities, one of the problems is the determination of responses based on a user's current state/context. Other problems include the determination of user location, user mobility, user activity, and user situation such as below:

- How to enable transparent distributed computing to continue operation across changing circumstances in a seamless manner.
- How to exploit changing environments to produce an awareness of the context of the location of use, the collection of nearby people and objects, accessible devices, and changes to those objects over time.
- How to evolve a new way of thinking for developing applications that are aware of the context to which the environment is sensitive and reactive to the presence of the user.

1.3 Scope of Study

This thesis is in an early stage of the study of Context-Aware Computing as a new paradigm in software engineering, a paradigm for the development of a software framework that is aware of the user context in the smart environment, which has capability to detect, assist, guide and respond to the presence of the user.

This thesis covers: 1. the design of the distributed architecture for Intelligent Environment, 2. a user location model including location resolution, 3. a user mobility model, 4. a user activity model, and 5. the provision of an intelligent response to the user.

A proposed symbolic (hierarchical) user location is also presented. The model is implemented in an indoor area, i.e. an Active Office.

Various fields of study of Context-Aware Computing have included on a superficial level and there is limited discussion of issues such as 1. localisation (the distributed data processing to find relevant local information when the user is on the move). This deals with the problem of discrete localisation, which resolves the user's location to a specific location, rather than the continuous localisation, 2. user navigation, 3. personalisation. This is a computing service based on context identity (user profile) and the presence of the user (presence awareness), 4. security and control service, 5. provision of user-context within the office, home and remote environments, and 6. any user activity in an outdoor environment.

1.4 Research Aims

The primary aims of this thesis are to prototype and evaluate complementary infrastructure (or tools) in an distributed environment to demonstrate how the provision of the user with ubiquitous access to information, communication and computation resources may be made as simply and easily as is possible in the Intelligent Environment. For this purpose, a scalable distributed context processing architecture is developed to manage and respond to rapidly changing aggregation sensor data in the Intelligent Environment.

Secondary aims include:

- The provision of the capability to survey the computing environment and react to changes to that environment.
- The provision of the capability to access or release resources without shifting the user’s mode of interaction from one resource to other resources.

The distributed systems architecture is developed to support portable and mobile devices/users in Intelligent Environments, leading to seamless connectivity in multiple environments. An environment model (Active Office) of Context-Aware Computing is developed to be ready-at-hand for everyday use.

This thesis focuses on user location model, user-context service delivery based on the user’s changing location (mobility), user activity model and Intelligent Environment response model as “proof of concept” and “proof of performance” of distributed architecture in an Intelligent Environment.

1.5 Methodology

The preliminary approach of this thesis is by understanding the concept of an Intelligent Environment, which required the understanding Context-Aware Computing. The aim of this preliminary study is to understand the concept of context, context awareness and connectivity between environment, location awareness, software architecture, context system interfacing, and precise and proximate sensors. The concepts of Ubiquitous/Pervasive Computing, Ambient Technology, Nomadic Computing, and Sentient Computing were also studied in characterising the Intelligent Environment.

From the understanding of the computing concepts above, the methodology to provide distributed support for Intelligent Environment was continued by the following:

a. developing a *distributed context processing system architecture* for an Intelligent Environment to present information and services to a mobile computer user.
b. designing models of user location, user mobility and user activity based on precise and proximate sensor data aggregation.
c. studying and analysing patterns of user mobility in an Active Office based on the user location history of sensor data. This offers the potential for the study of user behaviour and the prediction of the most probable user location when precise and proximate sensors cannot provide a user's location.
d. studying how the environment can provide intelligent responses to the presence of the user.

1.6 Contributions

This thesis's principal contribution is the development of Distributed Support for Intelligent Environments to solve important problems in Ubiquitous/Pervasive Computing, Ambient Intelligent Computing, Nomadic Computing and Sentient Computing areas, i.e. to enable transparent, distributed computing, and to allow continued operation in a seamless manner across changing circumstances, and contributes to the art of evaluation such systems.

The study further contributes to:

- The generic construction of distributed context processing architecture for an Intelligent Environment. The Intelligent Environment system would require the distributed architecture of the context-service system to be a globally scalable, locally efficient, secure service by identifying users and providing user context in the office environment.
- The solution of the problem of symbolic (hierarchical) user location by proposing the use of ηk-Nearest Neighbour in instance-based learning methods to estimate current-user location based on WiFi's signal strength and signal quality.
- Models of location awareness, by proposing a model that fuses the use of precise and proximate location sensors to estimate, to find and to detect 'current' user (or object) location and to predict 'future' user location based on user location history.
- Models of user mobility which is based on user mobility patterns and can be used for predicting future user movement based on *regularity* of user mobility and a proposed *pattern of accuracy* to adjust the degree of the regularity to actual user mobility.
- Models of user activity with the influence of context identity and context location as a practical characteristic to deduce the user situation.
- Methods for handling a large number and variety of fixed and proximate sensor data using a spatio-temporal data base approach.
- The design of a context layer architecture as an integrated part of DiCPA architecture for the context-aware environment that has the capability to respond physically to user activity.
- The provision of proof of the concept of how an environment response can be delivered to the user.

- The evaluation strategy for Context-Aware Computing to be used as a standard evaluation for better understanding the quality of Context-Aware Computing systems in objective and meaningful measurement.
- The changing of a paradigm: provision of service directly to where the user is located. The current paradigm is delivering service without knowing the user location and the new paradigm is the delivery of service directly to user location based on the profile of the user.

1.7 Outline of the Thesis

This thesis has ten chapters, the remaining chapters are organised as follows:

Chapter 2 presents the background of Context-Aware Computing. This includes; the relation from '*context*' to an Intelligent Environment; the relation between Intelligent Environments and Ubiquitous Computing/Pervasive Computing, Ambient Intelligent Computing, Nomadic Computing, and Sentient Computing. This chapter also presents the characteristics of the Intelligent Environment and related work in the Intelligent Environment, user mobility and user activity.

Chapter 3 develops a distributed architecture, DiCPA, which is a Distributed Context Processing Architecture for Intelligent Environments. The Merino Service layer, and context layer architecture are also explained in detail and include Intelligent Environment domain, Intelligent Environment repository, Intelligent Environment resolution, resources manager and knowledge-based context.

Chapter 4 develops an understanding of Location Awareness in the Intelligent Environment. The first step is to investigate user location categories based on precise and proximate sensors, the self organising approach for user location by using Bluetooth and WiFi's signal strength and signal quality. Based on this investigation, a User's Precise, Proximate and Predicted Location model is developed.

Further understanding of location awareness in the Intelligent Environment is the estimation of symbolic user location that will be presented in **Chapter 5** and User Mobility Model in **Chapter 6**.

In **Chapter 5**, to estimate Symbolic User Location, the use of Instance-Based Learning Methods using ηk-Nearest Neighbour is proposed in a machine learning algorithm for location awareness. The algorithm has been compared to other ηk-Nearest Neighbours and shows promising results. Since the quality of the training data-set has direct impact upon the estimation of user-location results, the Boolean MaxMin algorithm has been developed to evaluate the quality of data-sets.

The User Mobility Mode, based on history sensor data of changing user location, is investigated in **Chapter 6**. The pattern of User Mobility can be used to predict current user location when precise and proximate sensor data is not available.

User Activity in the Intelligent Environment is presented in **Chapter 7**. Studies are done of User Location History, User Identification and Authentication, User Profile, User Terminal and Access Network Characteristic, and Service Adaptation to User Environment.

Chapter 8 brings it all together, to provide intelligent responses in the smart environment. The intelligent response can be delivered based on user being (presence), user location and user activity. This chapter discusses how the environment recognises the user being (presence) in the environment, which requires user identity and location identity, followed by user location which leads to user mobility, then user activity which leads to user situation, and finally the response of the environment to the user which leads to action from the environment. The intelligent response is based on user patterns in the various precise and proximate sensors data. The problems in the Intelligent Environment are two-fold: every sensor has its own format and the volume of sensor data grows very quickly. To solve these problems, standardised formatting of sensor data and the fusion of sensor data using a spatio-temporal database are proposed in this chapter. Fusing sensor data as evidence of interaction between users and environment, can be used to develop a basis knowledge for the environment to provide response to the user.

Chapter 9 describes how this work is evaluated and it is also argued that good quality research for Context-Aware Computing requires *core content* and provides *iterative evaluation* in design process and product. This chapter also discusses the *software product quality metrics* for the evaluation of the Context-Aware Computing application which is influenced by two aspects: the technological and the social aspects, and two dimensions: the user and environment dimensions.

The thesis closes with a summary of *proof-of-concept* and *proof-of-performance* of this work, followed by conclusions and future research, which is presented in **Chapter 10.**

Chapter 2

CONTEXT-AWARE COMPUTING BACKGROUND

This chapter presents a review of the previous research done in areas that contribute to the understanding of "context" and Context-Aware Computing and which have wide acceptance in the area of Ubiquitous Computing/Pervasive Computing, Ambient Intelligent, Sentient Computing and Nomadic Computing. This chapter also describes the relations from "context" to an Intelligent Environment and reviews prior and related work in an Intelligent Environment. At the end of this chapter the background research on the current development of Active Office, User Mobility and User Activity and the evaluation in the Intelligent Environment are also presented.

2.1 A Brief of Context

The term "context awareness", was first introduced by Schilit and Theimer (Schilit and Theimer 1994). Their definition of "context" is "the locations and identities of nearby people and objects and changes to those objects". This definition is useful for mobile computing. It defines context by examples, and thus is difficult to generalise and apply to other domains.

Winograd points out that context are composed of "con" (with) and "text", and that context refers to the meaning that must be inferred from the adjacent text. Such meaning ranges from the references intended for indefinite articles such as "it" and "that" to the shared reference frame of ideas and objects that are suggested by a text (Winograd 2001). Context goes beyond immediate binding of articles to the establishment of a framework for communication based on shared experience. Such a shared framework provides a collection of roles and relations with which to organise meaning for the phrase.

Other researchers have defined context in terms of the situation and user activity. Cheverst et al. describes context in anecdotal form using scenarios from a context-aware tourist guide (Cheverst, Davies et al. 2000). Their work is considered as one of the early models for context-aware applications. Pascoe defines context as a subset of physical and conceptual states of interest to a particular entity (Pascoe 1998). This definition has sufficient generality to apply to a recognition system. Dey reviews definitions of context and provides a definition of context as any information that characterises a situation related to the interaction between humans, application and the surrounding environment. Situation refers to the current state of the environment. Context specifies the elements that must be observed to model a situation. An entity refers to a person, place, or object that is considered relevant to the interaction between a user and an application, including the user and applications themselves (Abowd, Dey et al. 1999; Dey, Abowd et al. 2001).

In this thesis, context is defined as predicate relation (see Section 8.2.1), focused on the relationship between two entities, i.e. an interactive user mobile computing device and surrounding elements of the environment.

2.2 Context-Aware Computing

Context-awareness is the use of context to provide task-relevant information and services interactively between a user mobile computing device and surrounding elements of the environment. A system is context-aware if it uses context to provide relevant information and services to the user, where relevancy depends on the user's task (Abowd, Dey et al. 1999).

The taxonomy of context-aware features is (Pascoe 1998):

- *Contextual sensing* (the ability to detect contextual information and present it to the user, augmenting the user's sensory system).
- *Contextual adaptation* (the ability to execute or modify a service automatically based on current context)
- *Contextual resources discovery* (to allow context-aware applications to locate and exploit resources and services that are relevant to the user's context).
- *Contextual augmentation* (the ability to associate digital data with the user's context – a user can view the data when he is in that associated context).

The key mechanism of context-awareness is (Abowd, Dey et al. 1999; Mantoro and Johnson 2003):

- Identity Awareness (Who), the awareness of the environment to user identity; included in these categories are user profile, persona, personalisation and user model.
- Location Awareness (Where), the capacity of the environment to recognise user location in open or closed space.
- Mobility Awareness, the capacity of the environment to use distributed systems and mobile communication to recognise the changing of a user's location to another location.
- Activity Awareness (What), the awareness - sensitivity and responsiveness - of the environment to the user's daily activity.

Many researchers offer contributions to the understanding of Context-Aware Computing. In 1997, Hull and Neaves et al. proposed that *Context-Aware Computing* is the ability of computing devices to detect and sense, interpret and respond to aspects of a user's local environment and the computing devices themselves (Hull, Neaves et al. 1997). In contrast, Dey et al. point out three current shortcomings in the field that include (Dey, Abowd et al. 2001):

- the notion of context is still ill defined
- there is a lack of conceptual models and methods
- no tools are available

Moreover they stated that there is no consensus in the field as to what "context" should include, and, as a result, it is hard to compare research directions and accomplishments across different researchers and groups. It is unlikely that a single definition will be accepted by all, but we can learn to understand the differences in approaches and how those differences shape the ways problems are addressed.

As a new software engineering approach, there is no common of conceptual model in Context-Aware Computing, Dey et.al proposed a "widget" model (Dey, Abowd et al. 2001), Hong and Landay proposed an "infrastructure" model (a "blackboard" model)

(Hong and Landay 2001), and other HCI researchers have a variety of models chosen from the understanding of tradeoffs among them.

Along with each model, new tools are being developed. Although they are all in early states of development, their tools can be expected to become part of the developers' context toolbox over the next few years.

From the description above, Context-Aware Computing, in this thesis, is defined as a new software engineering approach in the design and construction of a context-aware application which exploits rapid changes in access to relevant information and the availability of communication and computing resources in the mobile computing environment.

2.3 Ubiquitous Computing and Pervasive Computing

The concept of Ubiquitous Computing is computers everywhere. In 1988, Mark Weiser first articulated Ubiquitous Computing (UbiHome 1996) as invisible computation, making many computers available throughout the physical environment, while making them effectively invisible to the user. Ubiquitous computing is held by some to be the third wave of computing. The first wave was many people per computer, the second wave was one person per computer. The third wave will be many computers per person. The three technical issues are power consumption, user interface, and wireless connectivity (Weiser 1991; Weiser 1998).

Gregory Abowd proposes three key principles in Ubiquitous Computing which are:

- transparent interfaces,
- awareness, and
- automated capture.

Transparent interaction techniques include handwriting and gesture recognition, freeform pen interaction, speech, computational perception, tangible user interfaces (using physical objects to manipulate electronic information) and manipulation interfaces (embedding sensors on computational devices to allow for additional modes of interaction). Context-awareness allows for rapid personalisation of computing services. Automated capture records everyday experiences and makes those records available for later use (Abowd 1999). Roger Pressman put a Ubiquitous Computing as new software challenges in creating software to allow machines of all sizes to communicate with each other across vast networks (Pressman 2005).

Satyanarayanan defines Ubiquitous Computing as pervasive computing (Satyanarayanan 2001). He states that after a decade of progress, many critical elements of Ubiquitous Computing that were exotic in 1991 are now viable commercial products: handheld and wearable computers, Wireless LANs, and devices to sense and control appliances. He examines the predecessors of pervasive computing, *distributed systems* and *mobile computing,* and identifies four new research thrusts: *effective use of smart spaces*, *invisibility*, *localised scalability* and *masking uneven conditioning* of environments. Masking uneven conditioning is the technique to reduce the degree of variation of the huge differences in the ''smartness'' of different environments - what is available in a well-equipped conference room, office, or classroom may be more sophisticated than in other locations - in the user's view.

Characteristics. The essence of pervasive computing vision can be characterised as the creation of environments saturated with computing and communication capability, yet gracefully integrated with human users so that it becomes a "technology that disappears" (Satyanarayanan 2001).

A pervasive computing system has to be *context-aware*. It must be cognisant of its user's state and surroundings, and must modify its behaviour based on this information. A user's context can be quite rich, consisting of attributes such as physical location, physiological state (e.g. body temperature and heart rate), emotional state (e.g. affective computing) (Picard 1995), personal history, daily behaviour patterns, and so forth.

2.4 Ambient Intelligence

Ambient Intelligence (AmI) is the vision that in the future technology will become invisible, embedded in natural surroundings, present whenever it is needed, enabled by simple and effortless interactions, attuned to all our senses, adaptive to users and contexts, and acting autonomously. High quality information and content must be available to any user, anywhere, at any time and on any device (Lindwer, Marculescu et al. 2003).

In 2001, the Information Society Technologies Advisory Group (ISTAG), proposed a longer-term perspective in preparation for the next European Community Framework Programme for Research and Technology Development. The aim is to improve quality of life by creating desired atmosphere and functionality via intelligent, personalised, interconnected systems and services that might be enjoyed by ordinary people by 2010 (ECIS-Directorate-General 2001; Lindwer, Marculescu et al. 2003; Loenen 2003).

Characteristics. AmI refers to electronic environments that are sensitive and responsive to the presence of people. It stems from the convergence of three key technologies: *Ubiquitous Computing, Ubiquitous Communication and Intelligent User-friendly Interfaces* (ECIS-Directorate-General 2001). It can be defined as the merger of two important visions: "Ubiquitous Computing" and "Social User Interfaces". Key characteristics of such environments are: ubiquity, awareness, intelligence and natural interaction (Loenen 2003).

AmI implies a seamless environment of computing, advanced networking technology and specific interfaces. It is aware of the specific characteristics of human presence and personalities, takes care of needs, and is capable of responding intelligently to spoken or gestured indications of desire, and can even engage in intelligent dialogue. AmI should also be unobtrusive, often invisible, everywhere yet invisible unless people need it. Interaction should be relaxing and enjoyable for the citizen, and not involve a steep learning curve (ECIS-Directorate-General 2001).

2.5 Nomadic Computing

In 1995, Kleinrock introduced Nomadic Computing. His idea focuses on a major shift to nomadicity i.e. *Nomadic computing and communication*. He defines nomadicity as the system support needed to provide a rich set of computing and communication capabilities and services to nomads as they move from place to place in a transparent, integrated and convenient form (Bagrodia, Chu et al. 1995; Kleinrock 1995; Kleinrock 2000; Kleinrock 2001). He recognised that users are nomads. The nomad is characterised by always being on the move - between different groups of friends, work rooms, conference rooms, meeting rooms and so forth. The user often finds himself decoupled from “home base” computing and the communication environment (Bagrodia, Chu et al. 1995; Kleinrock 1995; Kleinrock 2000; Kleinrock 2001). Weiser mentioned that access to computing and communications is necessary not only from one’s “home base” but also while one is in transit and when one reaches one’s destination (Weiser 1991). Moreover, one may have more than a single “home base”, in fact, there may be no well-defined “home base” at all (Bagrodia, Chu et al. 1995) or it may be that there is a “virtual home base” on their device so no matter where the users may be physically they are still in one place digitally (Satchell and Singh 2004).

Kleinrock proposed a shift from a “connection-centric” view to a “service-centric” view (Kleinrock 2000) in which his focus is on providing services over broadband pipes.

Characteristics. Kleinrock also stated that desirable characteristics for nomadicity include independence of location, motion, computing platform, communication devices, and communication bandwidth with widespread presence of access to remote files, systems, and services. The notion “independence” refers not only to the quality of services but also to the perception of a computing environment that automatically adjusts to the processing, communication, and access available at the moment (Kleinrock 2000).

2.6 Sentient Computing

Sentient Computing can be understood as a synonym of similar terms found in the research literature, such as Context-Aware Computing (Ipina 2000; LCE 2002), pervasive computing, invisible computing or intelligent spaces (LCE 2002). This term is adopted to stress the use of sensors as a means to add perception, i.e. capability to see or hear, to computer systems (LCE 2002).

Sentient computing is the proposition that applications can be made more responsive and useful by observing and reacting to the physical world (Hopper 1999). The goal of sentient computing is how to deliver benefits to users, enabling them to interface directly to devices and express complex configuration requirements in a simple way (Hopper 1999).

It stems from the basic idea that to make computerised services pervasive in our lives, the devices providing such services must be given perception, i.e. the capability to recognise (see or hear) what entities are around them, what those entities are doing, where they are and when something is happening. Only then it is possible for the underlying computing system to undertake suitable actions to match end user expectations. For example, when a user enters his office, depending on the time and day of the week, the sentient environment could automatically initiate the playback of a specific kind of music. If a colleague then later comes into the room, the music volume could be automatically adjusted to facilitate communication. In order to do so, Sentient Computing must collect data from a variety of sensors in the environment and act upon those stimuli according to previously specified user preferences (Hopper 1999; Ipina and Lo 2001).

Many of the goals of Ubiquitous Computing are encompassed by the challenge to create a sentient building on a large scale (1000-10,000 people). Such a building would be equipped with sensors, with associated processing machinery integrated into systems that simplify and enhance its everyday use. This poses a grand challenge, because the construction of mobile and ubiquitous systems on this scale and with this degree of complexity and universality, has many unresolved problems (Harle and Coulouris 2002).

AT&T's Sentient Computing project (Hopper 1999) aims to replace human-computer direct interaction (through mouse or keyboard) by enabling users to instead interact with their surrounding spaces based on the precise location and orientation provided by the Active Bat Location System (Harter, Hopper et al. 2001). This project has been explored with several sophisticated sentient applications.

Sentient Computing is more than trying to realise Ubiquitous Computing vision. It is an intuitive physical action approach that initiates an appropriate response by an underlying computer system in which location and status data extends throughout the physical environment (Hopper 1999; Ipina and Lo 2001).

The challenge in sentient computing is to determine the appropriate meeting point between the physical notation of space and the logical constructs of our computer system (Hopper 1999). Sentient computing needs to make computer systems aware of the physical environment – shifting part of the onus of understanding from user to machine. Awareness comes through sensing, and that implies the need for appropriate sensor technologies to collect status and location data. Applications can be aware of their physical environment. They know where people and devices are and what devices can do. Sentient Computing also needs systems that react to users' actions with no perceptible delay, necessitating the updating of spatial information in real time, or near-real time (Hopper 1999).

A Sentient Computing system, as an application that shares the user's perception of the environment, uses wireless radio transceiver sensors to maintain a software model of the location of a set of laboratory users and objects. This model is then used for a number of intelligent "follow-me" applications. L´opez de Ipina (2000/1) defines sentient systems as systems that respond to stimuli provided by sensors distributed throughout the environment, by triggering actions that are appropriate to the changing context of the user (Ipina 2000; Ipina 2001)

Fitzpatrick, Bregel et al. proposes a sentient object which is an entity that consumes software events, and reacts by attempting to change the state of the real world via some hardware device (Fitzpatrick, Biegel et al. 2002). The possible applications of sentient objects are both numerous and diverse and include such areas as intelligent vehicles, smart buildings and mobile robotics.

In order to promote the use of the Sentient Computing paradigm in living spaces, it is not sufficient to make computing systems aware of the user's presence, movement or precise location, it is important that these systems are provided with the capability to automatically activate services on behalf of the user when the appropriate contextual conditions are met (Ipina and Lo 2001).

Research teams across the world have produced sentient spaces (Hopper 1999) encompassing one or two prototype rooms that detect the locations of people and devices and sense lighting, sound, temperature and other signals. Handheld or wearable computers with attached sensors and wireless connectivity provide users with continuously updated contextual information relevant to their current activity. Appliances and everyday artefacts sense and respond to changes in their human and physical environments. To expand to larger scale deployments and environments populated with non-specialist users requires a coherent effort from the Computer Science community to develop robust, convenient and efficient methods for the use of such sensory data and its integration with the digital environment (Harle and Coulouris 2002).

Characteristics. Sentient Computing concerns the ability of computing systems to *detect*, *interpret*, and *respond* to aspects of the user's local environment (LCE 2002). It uses sensors and resources status data to maintain a model of the world which is shared between users and applications. The goal is to augment computer applications with knowledge of the real world, allowing aspects of the environment to affect an application. The system employs sensors distributed throughout the environment to capture and maintain a detailed model of the real world and make it available to applications. Applications can respond to the environment and autonomously change their functionality - without explicit user intervention - based on observations such as who or what is around them, what they are doing, and where they are when something happens (Hopper 1999; AT&T-Lab-Cambridge 2001; LCE 2002).

2.7 Intelligent Environment

According to Weiser's vision, an Intelligent Environment is an environment with rich and rapid computing processing, which is highly aware of its inhabitants (Weiser 1991). The physical spaces of an Intelligent Environment are enhanced by computing capabilities to act intelligently: to observe, interact and react to inhabitants in a meaningful way. An Intelligent Environment understands human reasoning, analyses behaviours and infers intensions. Intelligent Environments actively collaborate with their inhabitants to assist them in making their surroundings more pleasant. Intelligent Environments even take decisions and execute actions based on their own learning. Intelligent Environments become integral participants in daily human activity.

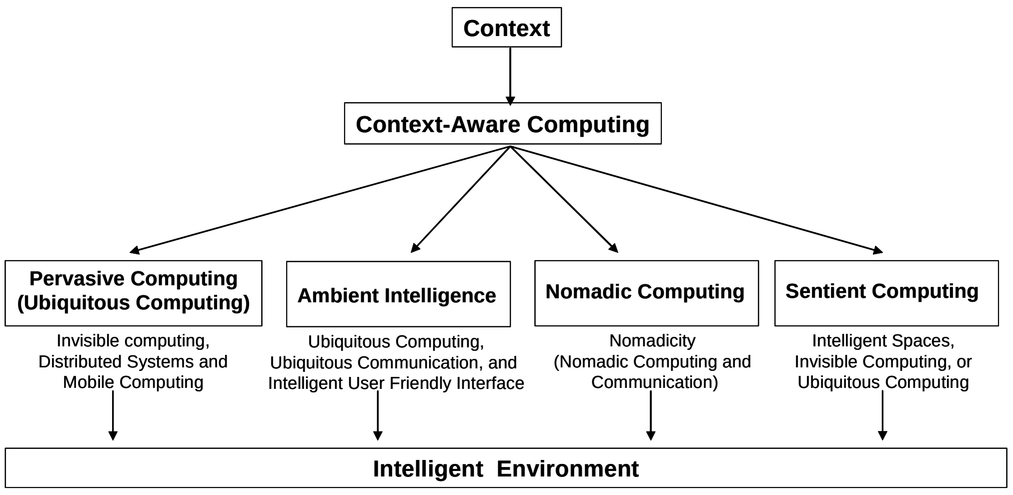

Context
Context-Aware Computing
Pervasive Computing (Ubiquitous Computing)
Ambient Intelligence
Nomadic Computing
Sentient Computing
Invisible computing, Distributed Systems and Mobile Computing
Ubiquitous Computing, Ubiquitous Communication, and Intelligent User Friendly Interface
Nomadicity (Nomadic Computing and Communication)
Intelligent Spaces, Invisible Computing, or Ubiquitous Computing
Intelligent Environment

Figure 2.1 The Relationship between Context and Intelligent Environment.

A critical element that Weiser anticipated, which has not yet been achieved, is the invisibility of pervasive systems. The ability of such systems to disappear into the background of everyday life is dependent on their ability to correctly interpret the state of the environment and act accordingly.

An Intelligent Environment, as shown in Figure 2.1, adopts the capability of Ubiquitous Computing, Pervasive Computing, Ambient Intelligence, Nomadic Computing and Sentient Computing. The basis of these computing technologies are Context-Aware Computing, which require the technology to be cognisant of the state and surrounding of the users and capable of determining what actions to take based on the context. Context-Aware Computing uses context to provide task-relevant information or services by giving computing devices the ability to detect and sense, interpret and respond to the computing environment.

Sections 2.1 to 2.7 are summarised by the user and technology/environment viewpoints and the characteristics of an Intelligent Environment, as shown in Figure 2.2. The user viewpoint is based on two variables; invisibility and capability/scalability, and the technology/environment viewpoint is based on three variables; network connectivity, embedded device and sensor/actuator.

The way in which an Intelligent Environment adopts the capability of Ubiquitous Computing, Pervasive Computing, Ambient Intelligence, Nomadic Computing and Sentient Computing from technology/environment viewpoint are as follows:

a. Network connectivity uses wireline and wireless connectivity, mobile devices, handheld connectivity and intermittent connectivity.
b. Embedded device uses network appliances embedded in natural surroundings, embedding intelligence into devices, into the environment (a room), and into portable device such as an Active Bat.
c. Sensor/Actuator can recognise user precise location and orientation (such as Active Bat location system), detect who and what in the Intelligent Environment, recognise the state and surroundings of the user, be unobtrusive, and sensitive and responsive to the presence of people.

The way in which an Intelligent Environment adopts the capability of Ubiquitous Computing, Pervasive Computing, Ambient Intelligence, Nomadic Computing and Sentient Computing from user viewpoint are as follows:

a. Invisibility:
    - Technology that disappears.
    - Invisible computation.
    - Some visible, many hidden away, but all networked.
    - Often invisible: everywhere yet invisible unless we need it.
    - Always available and always on.
b. Capability and scalability:
    - Present whenever it is needed, enabled by simple and effortless interactions, attuned to all our senses, adaptive to users and context and autonomously acting.
    - Physical location, physiological state (e.g. body temperature and heart rate), emotional state, personal history, daily behaviour patterns.
    - Responding intelligently to spoken or gestured indications of desire.
    - Independence of location, motion, computing platform, communication devices, and communication bandwidth with widespread presence of access to remote files, systems, and services.

	Technology/Environment View			User View		Characteristics
	Networked connectivity	**Embedded Device**	**Sensor/ Actuator**	**Invisibility**	**Capability and scalability**	
Pervasive Computing (Ubiquitous Computing)	Mobile devices. Wireline and Wireless	Portable devices.	Recognise user's state and surroundings. Unobtrusive.	Technology that disappears. Invisible computation. Some visible, many hidden away, but all networked.	Physical location, physiological state (e.g. body temperature and heart rate), emotional state, personal history, daily behaviour patterns	Computer everywhere (spread out), Computing and Communication on the move
Ambient Intelligence	Wireline and Wireless. Seamless.	Network Appliances embedded in natural surroundings. Embedding intelligence into devices.	Sensitive and responsive to the presence of people. Unobtrusive.	Often invisible: everywhere yet invisible unless we need it.	Present whenever it is needed, enabled by simple and effortless interactions, attuned to all our senses, adaptive to users and context and autonomously acting. Responding intelligently to spoken or gestured indications of desire.	Embedded in the environment. Ubiquity, awareness, intelligence and natural interaction. User empowerment.
Nomadic Computing	Intermittent connectivity. Wireline and Wireless.	Embedding to the nomadic environment (a room)	Detect who and what in the nomadic environment.	Invisible, ubiquitous, always available, and always on.	Independence of: location, motion, computing platform, communication devices, and communication bandwidth with widespread presence of access to remote files, systems, and services	The nomad moves in a transparent and convenient form. Shift from a "connection-centric" view to a "service-centric" view.
Sentient Computing	Wireless radio transceiver. Handheld wireless connectivity.	Portable device (Active Bat).	Recognise user precise location and orientation (Active Bat location system).	Invisible computing.	Large scale capability (1000-10,000 people) in a Sentient building. Enabling users to interact with their surrounding space based on the precise location and orientation. Meeting point between physical notion of space and the logical construct of the computing environment.	Targeting sentient object such as: intelligent vehicles, smart building and mobile robotics. To augmented computer applications with knowledge of the real world.

Figure 2.2 Intelligent Environment Characteristics

- Large scale capability (1000-10,000 people) in a sentient building.
- Enabling users to interact with their surrounding space based on the precise location and orientation.
- Meeting point between physical notion of space and the logical construct of the computing environment.

Characteristics. An Intelligent Environment can be characterised as a computing environment in open (outdoor) and closed (indoor) spaces with significant processing done by various fixed and mobile sensors, and which at least contains an Intelligent Environment repository, a resources manager, a knowledge base and a global/local resolution object naming scheme. The Intelligent Environment has the capability to recognise the user's location, activities and the social context defined by the presence of other people.

As in an Intelligent Environment, most of the significant process work is done electronically by fixed/precise and proximate sensors while the user is on the move. This situation allows the user to move (nomad) in a transparent and convenient form and shift from a "connection-centric" view to a "service-centric" view. As a consequence, computing and communication is also on the move, the computer in an Intelligent Environment should be everywhere (spread out) and always on, sensors and smart sensors could be embedded in the environment. Within these type of environments, users can be empowered, they are able to access the Intelligent Environment in ubiquity (apparent existence everywhere at the same time), awareness, intelligence and natural interaction.

2.8 Prior and Related Work in an Intelligent Environment

Since 1992, Mozer have been developing intelligent behaviour in an entirely adaptive environment in the University of Colorado (Mozer 1999). In 1996, while Coen were starting to build intelligent rooms in the MIT AI Lab. (Coen 1998; Coen 1999), in 1997 Kidd built the "aware home" in the Georgia Tech (Kidd, Orr et al. 1999), and at the end of 2000 Cisco Inc. built a broadband internet home (ihome) (Cisco-Inc. 2000).

Mozer's focus is on adaptability, building intelligence into various sensors (such as thermometers) and effectors (such as a heating and ventilation systems) so that it can adapt to the preferences of the house residents. One of his important questions is how intelligence can infer the preferences of its residents, such as by learning patterns in its occupants' behaviour, to adjust the setting of lights, ventilation and thermostats (Mozer 1999).

Much of Coen's efforts centre on the two intelligent rooms he built in the MIT AI Labs dealing with the integration of various sensing modalities such as vision and speech (Brooks 1997; Coen 1998; Coen 1999). In the aware home, Kidd focuses on finding frequently lost objects such as keys, wallets, glasses and remote controls using small radio-frequency tags attached to each object, and how to track the object using long-range indoor positioning systems. The aware home allows support of "everyday cognition" for the elderly using sensing technology (Kidd, Orr et al. 1999).

The Broadband Internet Home built by Cisco Inc. is essentially a home with broadband connections that provided an application to monitor and control the house from virtually anywhere, anytime, over the internet (Cisco-Inc. 2000). It does not use context-aware applications.

Abowd (1996) and Flanagan (1999) showed it is not only offices and homes that can be intelligent, but also rooms with large numbers of people such as lecture halls/classrooms and conference venues. Abowd and Atkeson et al. made substantial efforts to develop an educational environment, such as Classroom 2000, at Georgia University, that automatically creates records linking simultaneous streams of information, such as what the teacher is saying, while a student is writing her notes on a digital pad (Abowd, Atkeson et al. 1996; Abowd 1999), whereas Flanagan focuses on automatically recognising who in the audience is asking a question, pointing the video camera to that position and person and using a microphone array to filter out sounds coming from elsewhere in a large lecture hall at CAIP Center, Rutgers University (Flanagan 1999). He divides the problem into two parts, identifying the spatial location of the sound source, and then extracting the desired sound signal out of the collection of sounds being received by the microphones. He points out how microphone and digital signal processing technologies are enabling the creation of such intelligent rooms.

2.9 Active Office: Action Office for Knowledge Worker

Office and home domains offer different significant behaviours for habitants. Office as workplace is a place of productivity and home is a place for enjoyment and relaxation, but now it becomes possible for different forms of work to be performed within the physical space of the home.

The "workplace" is synonymous with a stereotyped notation of the "office". Closer analysis of the office as a container for work itself is not fixed (Churchill and Munro 2001). For example, in 1904 when Frank Lloyd Wright's Larkin Building was completed in Buffalo, USA, it was in many people's opinions the epitome of the concept of the office of that era. It represented the state of the art in special-purpose, design spaces for knowledge workers. The space was laid out efficiently as *an assembly line of the document*.

In this setting, workers were not mobile, not even locally. Documents were mobile and mobility of the worker was confined to supervisors.

Later in the 1960's Robert Probst proposed the Action Office, offering workers "power over the walls", where office interiors could be redesigned by moving walls and furniture as the need arose. Similarly, 1960s Germany, the Schnelle brothers introduce Burolandschaft, offering moveable furniture for the landscape office with information flow that led to the 70s cubicle culture and cube farms.

In the 1990s "hotdesking" arrived as furniture once again become immobile, but its "ownership" was highly mobile, with people and personal artifacts barely "touching the ground".

Work often takes place "out of the office". Much work takes place in other non-official locations. Some people had home offices, while others talked of appropriating space in an ad hoc manner. An example is "the corner of the kitchen table", a location that feminist designers have long noted represents a place where the domestic zone (the home) becomes an economic zone (a place of work). Work also possibly takes place in public places, a central issue that arises is the unpredictability of the environment and how technology including ubiquitous/pervasive computing technology can fit into this situation.

Technologies like the elevator affected the structure of the workplace, making the skyscraper possible and making for great concentrations of workers in one place. Further, innovations such as lighting and air conditioning affected the hours people were able to work effectively and comfortably.

Nowadays, communication technologies have affected the landscape of interaction, making communications between distant locations possible. Communication technologies like wireless technologies affected the way in which work gets done in terms of information flow without any wiring. Computer technologies and network infrastructures that have taken work out of a single place and into multiple places are one target of invisible computing.

In an Active Office, the worker will be mobile while information is available everywhere and every time in the office. The user also can be seen logically in several locations. The user mobility manages to provide relevant information at the right time and in multiple places. Our approach is by understanding user mobility patterns, understanding user tasks which lead to user activity in the corridor of Active Office criteria. The Active Office has three criteria i.e. high productivity, efficiency and comfort. The questions to designers are:

- What kind of working environment is it that the knowledge worker is expecting?
- How do the workers interact with their physical environments?

2.10 Related Work in User Mobility

In the current literature, the term 'user mobility' can be found in three research areas; 1. Immersive Virtual Environment, 2. cellular phone wireless mobile networks and 3. Intelligent Environment including nomadic computing, ubiquitous and pervasive computing. User mobility in this thesis has only a small relationship to user mobility in an Immersive Virtual Environment, which deals with how the users control their viewpoints to move in virtual space (Frees and Kesser 2003). This study also has only a loose relationship to user mobility in cellular phone wireless mobile networks, which enable a mobile user to communicate with others regardless of location. User mobility in this area has more mature technology than in the Intelligent Environment area. This cellular environment uses user mobility information as follows:

- assisted user mobility management (traffic routing) (Shen, Mark et al. 2000; Kravets, Carter et al. 2001),
- managed network resources, such as resources allocation, call admission control, congestion and flow control (Bellavista, Corradi et al. 2000; Shen, Mark et al. 2000),
- predicted user mobility (Liu, Bahl et al. 1998; Chan and Seneviratne 1999; Akyildiz and Wang 2004),
- user mobility patterns or movement pattern schemes (Zonoozi and Dassanayake 1997; Chan and Seneviratne 1999; Cayirci and Akyildiz 2002)
- User cell change processes in micro-cells and macro-cells (Liu, Bahl et al. 1998; Cho, Chung et al. 2000; Del-Re, Fantacci et al. 2000)

Most research into user mobility in the Ubiquitous Computing area focuses on the problem of host mobility, to allow the user access to the same service while moving. However, to do this the user needs to carry the same mobile host. Cui et al. argue that this is only a special case of user mobility (Cui, Nahrstedt et al. 2004). In the Intelligent Environment, user mobility not only includes host mobility, but also includes the case where a user is free to switch from one host to another as well and move from one location or computing environment to another. Further, in the case of user mobility, a user is free to access his personalised service anytime, anywhere, through any possible mobile or fixed device (Sousa and Garlan 2002).

While the use of movement prediction seems to be a promising approach for improving the efficiency, reliability and adaptability of wireless networks, the actual user mobility client patterns are not yet well understood (Chan and Seneviratne 1999).

In the literature, current approaches to user mobility in ubiquitous computing are based on one of five techniques, none of which fully achieves the ubiquitous computing goal. One approach is to support as much of a user's computing needs as possible on a mobile machine. A second approach is to compute via remote access to a computing server that stores a user's personal state and preferences, as much as can be done with VNC or XDM (X-terminals). A third approach is to provide standard applications that are ported to and installed in all environments. Those applications are extended to become aware of user intention and mobility. A fourth approach provides standard virtual platforms (such as the Java Virtual Machine) that enables mobile code to follow the user as needed (Sousa and Garlan 2002). The last approach is to provide user access from one terminal to another actual terminal, which is not a virtual terminal such as VNC does, in his working environment. This fifth approach requires the system to authenticate a user entering the system and to organise a user's working environment accordingly.

There are two problems with these approaches. First, since to some degree the assumption of homogenous computing baseline is used, this cannot take full advantage of the diverse capabilities of each environment, such as external displays, processors, and I/O devices. Second, the lack of ability to handle dynamic variations to capabilities and resources in the context-aware environment without overburdening the user with manual tuning and reconfiguration (Sousa and Garlan 2002; Cui, Nahrstedt et al. 2004).

A user mobility use mostly in wireless environment, which the availability of large bandwidth, low error rates and always-on connectivity exists as it has in a wired environment. Mobile user devices need to quickly detect and adapt to drastic changes in the characteristics and available resources of the wireless environment. In this study, middleware technology is considered for this purpose, as intermediary to providing higher-level network services and abstract service environments for distributed applications.

The middleware technology supports transparent service to users. This transparency builds a new form of awareness of an environment that allows the execution context and adapts to middleware behaviour. Unfortunately in executing resources and execution environments from direct application participation often results in premature termination of the problematic application if available resources are depleted. An alternative approach is to develop an event-driven mobile middleware that supports a degree of flexibility by allowing direct participation of applications in adapting to changes in resources. In the formulation of a context-aware middleware, suitable control mechanisms were required to directly participate in resources adaptation in response to the dynamic operating environment (Chan, Chuang et al. 2004). The scripting of an event service including the declarative language can be developed based on XML, which supports synchronous event call-backs, to express the conditions that are bound to a specific event in order to build a composite event from a set of primitive events.

The early event model has been introduced on agent-oriented software engineering, such as ROADMAP (Juan, Sterling et al. 2002), Gaia (Wooldridge, Jennings et al. 2000; Juan, Pearce et al. 2002), Prometheus (Padgham and Winikoff 2002). In event model, the object abstractions are on the flow and the processing of events is applied across different levels leading to the ultimate notification of events to the service objects. For example, Chuang et al. developed event platform in WebPADS (Chuang, Chan et al. 2002). When the WebPADS client starts executing, a default-service chain is created, which is based on the description in XML configuration. The service-chain map is attached to an environment monitor, which regulates the time and conditions that determine when reconfiguration of the service is to take place (Chan, Chuang et al. 2004), which can be used for monitoring user mobility.

The user mobility problem in Ubiquitous Computing has significant challenges in developing Active Office, for example, developing infrastructure with a variety of wired and wireless sensors, fusing sensor data using a spatio-temporal database (see Section 8.4.1), and the use of machine learning (see Section 5.2) for a variety of user locations and user activities, the use of event-driven mobile middleware for a variety of user mobility. In developing context aware systems, it also challenges because of the use of already existing devices (old sensors and networks) and new embedded devices, to develop context-aware applications, which require toolkits designed to enable maximum capability of the devices/sensors. The toolkit design should follow the requirements of the context aware applications, which are:

a. ease of deployment
b. network and sensor scalability through sensor fusion rather than in precision
c. enabling wider acceptance through better design for user needs
d. increasing human trust in the system's care for user privacy and security.

This study contributes to user mobility in the Ubiquitous Computing area, especially in indoor space, which is a user mobility model based on user mobility patterns in the history of the predicted and proximate users' locations in an Active Office. The location history that was collected used wired and wireless sensors and scalable context processing.

2.11 Related Work in User Activity

In the current literature, researchers who work in the area of user activity fall into three categories, they: 1. develop equipment using wearable devices to be worn by the user to sense user activity and recognise user location, 2. study user behaviour in the workplace area and home, and 3. develop a system/device to equip the environment. The tree structure of research categories in the area of user activity is described in Figure 2.3.

Firstly, in the area of wearable devices, it is possible to determine the user's location using a dead-reckoning method to detect the transition between preselected locations and recognise and classify sitting, standing and walking behaviours (Lee and Mase 2002).

Secondly, in the study of user behaviour, user activity changes in work and society are impinging on the place where the user works and how the work gets done. For example, the increasingly international nature of work has led to a growing amount of travel despite the use of advanced collaboration technologies (Churchill and Munro 2001). It has been argued that many more people are experiencing a blurring of the division between 'home' and 'work' domains as different forms of work become possible within the physical space of the home.

While Koile proposes activity zones to construct an activity area, Crabtree introduces communication places. Activity zones have a physical form – e.g. wall, furniture – which partition places into zones of human activity and places of communication that are familiar in the home as areas of production, management and consumption of communication. Activity zones were constructed by tracking systems to observe people's activities over time. Crabtree considers three different properties: ecological habitats, activity centers and coordinate displays, where Activity centers are places where media are actively produced and consumed and where information is transformed (Crabtree, Rodden et al. 2003; Koile, Toolman et al. 2003).

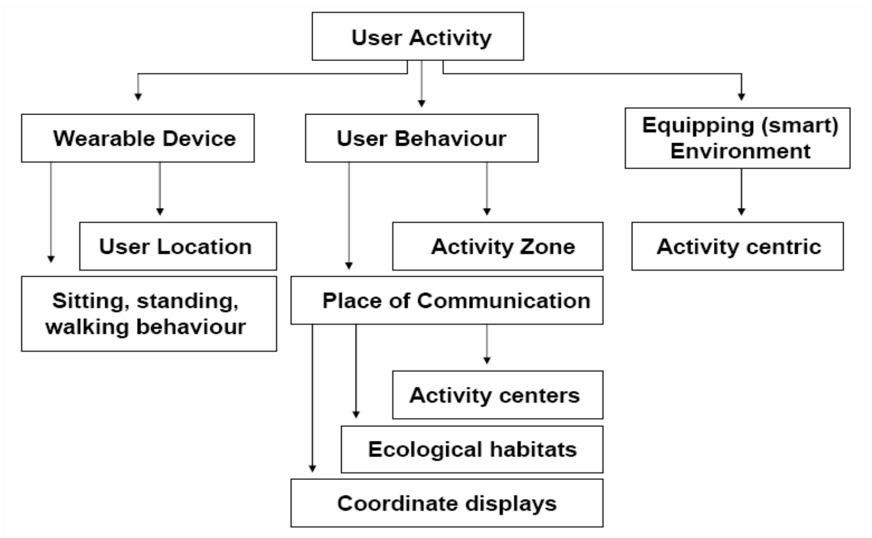


Figure 2.3 Research Categories in the Area of User Activity

Thirdly, in the area of developing a system/device to equip the environment, Prekop and Burnett developed an Activity-Centric context (Prekop and Burnett 2002), i.e context-aware applications that are capable of supporting complex and cognitive user activities in a smart room. Mantoro studied user mobility based on user location leading to user activity in the Active Office (Mantoro and Johnson 2003). Currently there are a number of smart environments already in use in research organisations, for example, MIT's Intelligent Room (Benerecetti, Bouquet et al. 2000), Stanford iRoom Project (Brown, Bovey et al. 1997), NIST's Smart Space Lab (Budzik and Hammod 2000), Georgia Tech's Aware Home project (Kidd, Orr et al. 1999) and ANU's Active Office (Mantoro 2003; Mantoro and Johnson 2003; Mantoro and Johnson 2003; Mantoro and Johnson 2004).

To provide a dynamic environment of located-objects, Schilit proposed Active Map Geographic Information to manage information about the relationship that exists between locations (Schilit 1995). In people's daily lives two kinds of spatial relationships are commonly used: containment and travel distance. Schilit mentioned that Euclidian distance between positions within a coordinate system are not suitable for human activity (Schilit 1995).

Related study in user activity also considers the social aspect of the user. This research is mostly study in user dimensions, instead of environment dimensions or technological aspects, such as the study of the use of time (Szalai 1972). Time use studies typically have a single focus: to study the frequency and duration of human activities (Stinson 1999).

According to Stinson, the use-of-time for Canada's telephone administration can be places into two categories, i.e. first *in places*, such as a respondent's home, a workplace, at someone else's home, at another place (include park, neighborhood); and second *in transit*, such as in a car (driver or passenger), walking, in a bus or subway, on a bicycle, other (airplane, train, motorcycle). Throughout the world, most of the currently used activity classification systems have evolved from the original structure developed by Alexander Szalai for the Multinational Time-Use Project of the 1960s. These activity codes are typically arranged into mutually exclusive behaviour groups that cover all aspects of human activity. These primary divisions of behaviour, which may be considered for the study of user activity in the Intelligent Environment, generally include:

- Personal care activities
- Employment related activities;
- Education activities
- Domestic activities
- Child care activities
- Purchasing goods and services
- Voluntary work and care activities
- Social and community activities
- Recreation and leisure
- Travel time

However, the recent technically advanced studies in Active Badge/Bat (Cambridge), Wearable Computing (University of South Australia), Cricket (MIT), and Smart Floor are also enabling the creation of such Intelligent Environments in capturing and understanding user activity (Thomas, Demczuk et al. 1998; Orr and Abowd 2000; Priyantha, Chakraborty et al. 2000; Harter, Hopper et al. 2001) These advances in technology to equip the environment have demonstrated the potential to observe user activity, but have also shown that these kinds of systems are still extremely difficult to develop and maintain (Hong and Landay 2001; Mantoro and Johnson 2004).

2.12 Evaluation in the Context-Aware Computing

Evaluation in the Context-Aware Computing area is essential to provide objective and meaningful measurement and understanding of performance. Evaluation assesses the processes of *analysis, design* and *the result/product* of new computing technology innovation and concerns all system components, especially *computing technology* and *human components*. Systemic evaluation should appear as a first step overhead activity for all components.

Context-Aware Computing currently lacks a systematic user-centered evaluation methodology for interactive systems, but many evaluation methodologies have been developed in the Human Computer Interaction (HCI) area and used within the community. Unfortunately Context-Aware Computing evaluation lacks the higher-level evaluation methodology that can increase overall understanding and help advance HCI research (Scholtz 2001).

The evaluation techniques of Context-Aware Computing are borrowed from other fields. In particular, context acquisition systems are often evaluated using methods known from Artificial Intelligence and Neural Networks, whereas context-aware user interfaces are evaluated using standard HCI methods. It is clear that the evaluation of sub-parts of a system does not necessarily provide an overall assessment of the system, as long as the essential components of sub-parts fulfil the requirements (Schmidt 2002).

As more new methods are introduced, the variety of alternative approaches and a general lack of understanding of the capabilities and limitations of each have intensified the need for practitioners and others to be able to determine which methods are effective in what ways and to what purposes. In reality, researchers find it difficult to reliably compare the evaluation methods (Hartson, Andre et al. 2003) because of the lack of :

- standard criteria for comparison.
- standard definitions, measures, and metrics on which to base criteria.
- stable, standard processes for usability evaluation and comparison.

There is no single standard for direct evaluation comparison, resulting in a multiplicity of different measurements used in the studies, capturing different data in different ways. Consequently, very few studies clearly identify the target criteria against which to measure the success of an evaluation being examined. As a result, evaluation comparison studies are not accurate, are reported as not complete or otherwise fall short of the kind of science contribution needed (Gray and Salzman 1998; Hartson, Andre et al. 2003).

Evaluation serves two purposes (Scholtz 2001):

- helping researchers better understand research issues and enabling researchers to measure progress in a field.
- produce a more focused effort to overcome particular difficult research problems and make research progress.

Evaluation in Context-Aware Computing targets how well systems can reason about context, especially in relation to user-centered evaluation. This evaluation includes how to deliver useful information to the user at the right time and in the right form, as well as how the user's cognitive load can be lessened by Context-Aware Computing for work, entertainment or education. If it is an entertainment task, does a context-aware system increase the user's enjoyment (Scholtz 2001) If the enjoyment is below the user's expectation the redesigning process should be repeated until it reaches the required specification. Thus, the evaluation needs to be iterative.

The evaluation of Context-Aware Computing may be in one or more of

- Computer technology.
- Usability.
- Usage.

- System and social implications.

The criteria for evaluation are not onerous. *Computer technology* can be evaluated for novelty; by its description, such as architectural, structural; or measured, as power, precision, robustness, reliability, battery life, cost and possibly performance, though description or measurement does not appear to be significant for most technologies at this stage in the area.

Usability in HCI and *usage* of acceptability or user responses are based on small samples in this discipline. A usability evaluation report on as few as six users is common. Sometimes, it is not thought necessary to do an extended social survey, a broad spectrum user study or double blind trials. The basic criterion for an acceptance evaluation is that more is done than simply present a new implementation with the implied statement "Isn't this great, take it or leave it".

Evaluating at the system level means taking a step back from project achievements and being critical of the quality, how well does it cover the area of the targeted application or infrastructure, what shortcomings does it have, what possibly unexpected bonuses? *System software qualities* could be composed of correctness, reliability, error tolerance, robustness, modularity, eases of extension, and ease of re-use. *External considerations* could be social implications, acceptability to users, economics, personal trust and security.

Context-Aware Computing systems are designed to develop the full potential "in context" especially to help users in certain situations and to provide relevant and useful information for particular tasks. The best option for the evaluation of such a system is where the evaluation occurs when the user is in context. The problem is that such a situation may not happen very frequently and forcing a situation to happen has the potential to change the usage experience, with the result not necessarily reflecting real usage in the situation. On the other hand, prolonging the experiment to make sure that the situation will occur is not practicable, as such a user study could take years (Schmidt 2002).

Most often the goal of the evaluation in a conventional system is proof of performance in computer run-time and space. Variables associated with the performance in a context-aware systems, include the time to complete the task and the number of errors while fulfilling the task. In Ubiquitous Computing when a system augments an environment, the evaluation goal includes the case of enabling a user to do new things or to make boring tasks more interesting or more pleasurable, the metric[1] is not directly measuring only the environment or user performance but also evaluating user usability as well as his social context. This is relatively different to the evaluation of system design in Structured Analysis and Design Technique (SADT) which focuses more on the breakdown of the bubble process and data modeling and not sufficiently on the evaluation of system usability.

When a task is made more interesting, exciting and pleasurable, evaluation depends on the subject doing the task, therefore the result will not be objective. In this case, qualitative evaluation is needed instead of quantitative evaluation. Multi Criteria Decision Making (MCDM) (Tabucanon, 1988;Triantaphyllou, 2000) can be considered as the best

[1] A standard of numeric measurement of the product and process especially in software development.

candidate for use in the evaluation, using a well defined group of subjects (selected users) to achieve a good insight evaluation.

In evaluating a ubiquitous system, understanding the evaluation goal is important in the preliminary stages of evaluation. The goals can be in qualitative evaluation of the concept, to demonstrate the feasibility of the concept, proof of the quality of the system design, to enhance user experience and user interface concepts. It can also be in quantitative evaluation, showing the ease of use, and proving the efficiency or stability of implementation. All these issues require different evaluation procedures. Identifying what should be evaluated before commencing the evaluation can assist in providing a good result. Unfortunately many developers don't use evaluation carefully because the evaluation methods are too time consuming, costly and the techniques are often intimidating in their complexity.

Chapter 9 discusses notions of "good quality" in Context-Aware Computing followed by evaluation criteria, metrics evaluation and usability evaluation for Context-Aware Computing. This evaluation section discusses evaluation for experimental for experimental in Context-Aware Computing followed by evaluation of the prototype, iterative evaluation of the design process and of the product/device, impact of user factor/characteristics; the damaged merchandise and the discount engineering concept in Context-Aware Computing.

2.12.1 Evaluation for experimental in Context Aware Computing

As a human creation (artefact), good quality research in Ubiquitous Computing is an important part of experimental computer science and engineering (ECSE). ECSE fundamentally is a synthetic discipline to study the phenomena that are entirely the product of human creation (artefact). This discipline concerns how research in ECSE is evaluated. According to Snyder et al. (1994), as a proof-of-performance for artefact, it should demonstrate that the artefact is better because it is objectively faster or consumes fewer resources. However, it may be harder to prove the greater worth of artefacts serving a proof-of-concept or a proof-of-existence role, because the advancement may be qualitative, as in increased functionality, where the production of computing artefacts depends on a substantial infrastructure. However, the evaluation for ECSE can be in one or more proving processes as follows (Snyder, Baskett et al. 1994):

1. *Proof-of-performance.*

   Proof-of-performance relates to the performance of the system itself, typically space (memory) and time (response time, throughput rates). An artefact acting in the proof-of-performance role provides an apparatus or test bed (prototype) for direct measurement and experimentation. The Artefact can be constructed and the result is usually quantitative.

   The artefact is usually the apparatus and better can mean more efficient. Efficiency metrics include higher speed, smaller memory requirements, less frequent disk references, and so on. Better in this sense is determined by direct measurement and is quantitative. Better also means more functional or enhanced functionality or more expressive.

The size and complexity of artefacts are constrained by the resources that can be invested in them - researcher time, funding, equipment, infrastructure, and so on.

2. *Proof-of-Concept.*
   An artefact acting in the proof-of-concept role demonstrates (prototype) by its behaviour that a complete assembly of components can accomplish a particular set of activities, behaviour that could not be argued simply by logical reasoning or abstract argument from first principles. This type of artefact is usually itself the subject of the research.

   Proof in terms of a demonstration/prototype is necessary. The working system, the artefact, is a witness "proving" that the concepts in at least one configuration are correct. Experimental computers are good examples of proof-of-concept.
3. *Proof-of-Existence.*
   An artefact playing the Proof-of-Existence role conveys the essence of an entirely new phenomenon. Because computation is synthetic, human creativity can produce phenomena never before imagined, which are often explained better by demonstration/prototype than by description.

"Good quality" in Context-Aware Computing must have interesting *core content* and *iterative evaluation* in both, *user and environment dimensions* including evaluation of the design for robustness of the context-aware system. The *social aspect* and interesting *computer technology aspect* can be considered as principles criteria of *core content* for "good research" in Context-Aware Computing. The *metrics evaluation* and *demonstrator* are the requirement for establishing the *proof-of-performance*, *proof-of-concept* and *proof-of-existence*. The design metrics evaluation for Context-Aware Computing discussed in Section 9.3. The use of prototype as demonstration can be the best proving for the good artefact. The evaluation of prototyping as demonstrator for Context-aware Computing will be described in Section 2.12.2. This study performs the evaluation of the prototyping in Section 9.5 and the proof-of-concept and proof-of-performance is discussed in Chapter 10.

2.12.2 Evaluation of the Prototype in Context Aware Computing

Communicating novel ideas and concepts can very difficult as there is often little common understanding of issue and problems between research communities and their non-technical and non-research counterparts. However, potential customer and ultimately buyer feedback at this early stage is very beneficial to the development of technologies. If there is no communication between potential users there is the serious risk that research will explore issues that are of no interest to anyone (Schmidt 2002). The prototype can be used to demonstrate the *core content* of computer technology in Context-Aware Computing area. The use of a prototype, a rapid prototype and demonstrations can be the best communication by which the researcher can show technology, and for potential users to imagine the technology in their daily lives, to assess its impact on everyday tasks, and discriminate between the novelty effect of having a new device and the added value of a new artefact.

Because of the abstract nature of technology, potential users may sometimes not fully consider their needs and requirements, instead of choosing to accept descriptions that are more from the world of science fiction than reality. Evaluation of the prototype is needed for this purpose even when functionality is not fully implemented.

Prototyping obviously can make life much easier, but it can also just delay tackling difficult aspects (Schmidt 2002) that require resolution. The advantages of prototyping are as below:

- A prototype provides a usage experience, similar to the system envisaged.
- By simulation of the real system, it has the potential to offer insight into many different aspects of user experience and usage.
- Useful in the creation of a final system, especially where the prototype adheres to real conditions i.e., power constraints, weight constraints, cost, size and to some extent includes the effect of future development.
- Compiling Results.
  The use of interviews, questionnaires and statistics are standard tools in the acquisition of outcomes of the evaluation of a prototype in a living lab experiment.

The disadvantages of prototyping are as below:

- Generation Incompatibility.
  Often a prototype develops gradually, as knowledge is gained, systems are improved or at least altered over generations. These changes lead to incompatibility between artefacts, i.e., makes program complexity worse, a problem that can only be resolved by updating the whole system.
- External Dependencies.
  When a system is used for a long period, different problems may occur, as a system is not isolated from changes to other parts of the system. This problem is beyond the reach of the developer to force such system changes. This problem may be solved by the re-implementing of a sub-system.
- Maintenance and Support.
  Prototypes are very often not maintenance free, even if the envisioned system will have this property. It needs people to maintain and support it. Maintenance is a time-critical issue, and it is important to get systems going quickly after a breakdown so that users will keep using the system.
- Incomplete System.
  A prototype is obviously not a complete system. However, it is useful to deploy working parts of a system in the living lab. This requires part of the system to be built in a way that it can run without the rest of the system, optimally providing some benefit, even in an incomplete state.

In the evaluation of prototypical systems, where the primary goal is to explore the implications of a new user experience and a new relationship between the user and the computer, the actual technology used in the production of the prototype is not of central interest (Schmidt 2002). This is assumed that at the point when such installations will be widely available the technology will have changed over time.

One of the important problems that remain beyond evaluation as well is how to extract valid knowledge beyond a single case that is useful in further development. Many published results are taken from a very specific prototype system and while results may be valid in most cases they are not reproducible (Schmidt 2002). Proving the validity of a more general claim, itself abstracted from a single case or from a few applications, is in many cases difficult or even impossible.

2.12.3 Iterative Evaluation of the Design Process and of the Product/Device

As mentioned in Section 9.2 that in the production phase of Context-Aware Computing, design errors require a redesign of the hardware and/or software and also redesign of the tool-kit and equipment involved in the production life-cycle. The programming complexity to develop the context aware application also requires a highly qualified and motivated staff and good working condition; otherwise software projects may be delayed. Therefore, the earlier the human-factor issues in the design can be addressed, the less the cost to be incurred in correcting any errors and misconceptions.

To encounter this problems, our approach are to develop two phases evaluation for Context-Aware Computing, which contains: design evaluation and product evaluation. Both evaluations have iterative processes involving design, design evaluation, redesign, product, and product evaluation (Figure 2.4). The major iterative design process can be in three stages: initial design, prototype and final design, with iterative product evaluation following product development life-cycle, which is introduction, product growth, mature stage, saturated stage, decline stage and vanish stage.

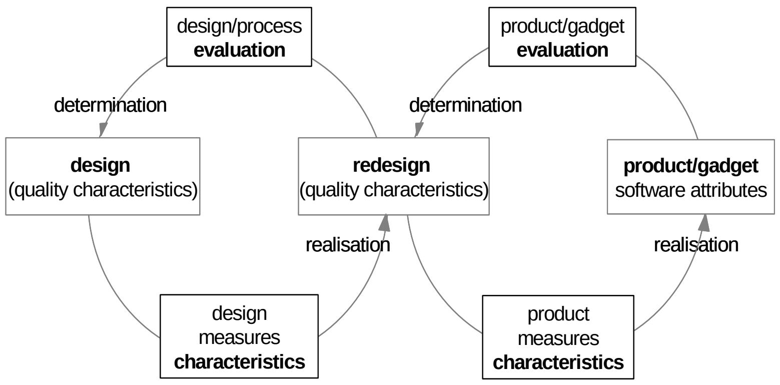


Figure 2.4: Iteration of the Design and Product Evaluation

The purpose of design evaluation is to avoid unwelcome surprises before the system goes into production. This is an iterative design, which begins with determination of quality characteristics for the design, is followed by measurement of design characteristics to define the degree of characteristic quality. This stage can be achieved with the involvement of the users/buyer/customer. When the system’s design is finished, design evaluation sees comparison of the determination and the realisation of quality characteristics. The design evaluation process continues in an iterative process until the user is satisfied with the realisation of the redesign.

The final stage of redesign is the determination of the quality characteristics of the product. The quality characteristics at this stage can be external characteristics from another product or environment that may have direct or indirect impact on the product. Then the process continues to the measurement of characteristics that define the quality characteristics of the product. Again, at this stage, the user/customer or buyer will have more involvement in the process. Section 2.12.4 shows how the user type has direct involvement in final design product.

When the product is ready, product evaluation begins by comparing the determination and the realisation of product quality characteristics. The product evaluation process continues iteratively until the user is satisfied with the realisation of the redesign. The purpose of product evaluation is the avoidance of unwelcome surprises in acceptance testing for both technical and user social aspects.

2.12.4 The Impact of User Factors/Characteristics on Context-Aware Computing Design

Evaluation processes, especially the software and device product evaluation processes, should deal with the user. The evaluation process includes user situation, user preference, user profile, as well as user mental effort in understanding and learning the software and device product. The user is the most important aspect to consider in a Context-Aware Computing product, because user characteristics have direct impact on the design of context-aware applications. The most common user characteristics are as follows:

- type of user (end user, remote user, disability user).
- experience (business process, automation, inexperienced user).
- knowledge level (education background).
- Age.
- Gender.
- number of users (individuals, groups).

In a situation where many of the users are inexperienced and have relatively low education backgrounds, the requirements for learnability in the design of context-aware application will be high. On the other hand, in a situation when many managers and users have relatively high levels of experience, the requirements for time behaviour (performance, response time) in the design of a context-aware application will be high.

2.12.5 Damaged Merchandise and Discount of Engineering

Gray and Salzman (1998) introduced the concept of "*damaged merchandise*", specific to concerns about the validity of evaluation comparison studies. This concept is a reference to the lack of attention given by researchers to rigorous experimental design for the evaluation of a software product.

They made the case that, when the results of a study that is not rigorously designed and executed according to the prescripts of experimental design methodology for a statistical approach are used to inform the choice of which usability evaluation to use, the consequence is damaged merchandise.

They made the point that numerous small problems can be found in research design and conduct. If these small problems were to have only small effects on the research area, they could be ignored. Unfortunately, these small problems have the potential to produce large effects from an experimental study. In Context-Aware Computing, this can bring into question what is acceptable to the research design and usability evaluation. A key concern is the issue of the use of the right measurements in comparing design and usability evaluations in the determination of effectiveness.

Some of the incomplete results in context-aware applications were sometimes understandable because researchers were using data that became available through means designed for other ends, i.e., usability testing small devices within a real development project where additional resources were not available to conduct a complete, scientifically valid experiment. These partial results have merit as indicators of relative usability evaluation values in a field in which complete scientific results are scarce.

In many engineering contexts, including usability for Context-Aware Computing, the concept of damaged merchandise is not always a bad thing. For example, most usability evaluation represents a conscious trade-off of performance for savings in cost. As long as "buyers" know what they are getting, and that it will be sufficient for their needs, they can often get a good "price" for damaged merchandise. This is a sound principle behind what has been called *discount engineering* (Nielsen 1994) and which has always been part of the legitimate difference between science and engineering.

However, as pointed out by Gray and Salzman (1998), damaged merchandise is far less acceptable in the realm of usability science, especially when found in the form of poorly designed usability evaluation comparison studies. There is certainly a need for more carefully designed comparison experiments, both to contribute to the science of usability and to provide practitioners with more reliable information about the relative performance of various usability evaluations as used for various purposes.

Some authors in the discussion sequel to the Gray and Salzman article, as compiled by Olson and Moran, have suggested that *some science is better than none*, and resource limitations that preclude complete scientific results should not prevent attempts at modest contributions (Olson and Moran 1998). These discussants have argued that this is especially true in a relatively new field in which any kind of result is difficult to come by. On balance, Gray and Salzman would probably caution that sometimes *bad science is worse than none*. However, as Lund pointed out in his commentary on Gray and Salzman, the danger may not loom so darkly to practitioners, making the case that practitioners will quickly discover if a recommendation is not useful (Lund 1998). In any case, the argument made previously, which applies to the acceptance of any merchandise, was based on the buyers knowing what they are, and are not, getting for their money.

2.13 Summary

Chapter 2 presents the background of this thesis. This chapter describes the relation from "context" to Context-Aware Computing and to Ubiquitous Computing, Pervasive Computing, AmI, Nomadic Computing and Sentient Computing and to Intelligent Environment.

Context is about predicate relations. As will be discussed in Section 8.2.1, context is defined as rich and rapidly changing predicate relations between objects (user and environment entity) that contain information relevant to the current local domain while an object (user entity) is on the move. This information is used to recognise the presence, location, mobility, activity and situation of the user entity and to respond and take action toward the environment entity.

In this thesis, a context-aware system is a system that uses context to provide task-relevant information and services interactively between a user mobile computing device and surrounding elements of the environment. These definitions of "context" and context-aware systems lead to the definition of Context-Aware Computing as a new software engineering approach in the design and construction of a context-aware application that exploits rapid changes in access to relevant information and the availability of communication and computing resources in the mobile computing environment.

To summarise Section 2.2 to 2.6; Context-Aware Computing is the basis technology of Ubiquitous Computing, Pervasive Computing, Ambient Intelligence, Nomadic Computing and Sentient Computing. All the capability of these technologies, such as networked connectivity, embedded device, sensor/actuator, invisibility, capability and scalability, are absorbed and adopted by an Intelligent Environment.

Prior and related works in an Intelligent Environment have been discussed, including the background and current development of Active Office, User Mobility and User Activity and the evaluation in the Context-Aware Computing area.

Chapter 3

DISTRIBUTED ARCHITECTURE FOR INTELLIGENT ENVIRONMENTS

This chapter describes distributed context processing architecture for an Intelligent Environment. The architecture contains at least an Intelligent Environment repository, an Intelligent Environment resolution, a knowledge base and a resources manager as an integrated part of a context awareness mechanism in an Intelligent Environment. How the scalable distribution of context and user model information might be managed in an Intelligent Environment is also described in this chapter. An implementation model of the architecture is also illustrated by providing automatic connection to an unfamiliar Intelligent Environment domain.

3.1 Introduction

The future of the Intelligent Environment is based on the premise that computing and sensing devices will become ubiquitous. Context-awareness mechanisms are the best way for computer applications to operate in such a computing-rich environment (Abowd 1999, Satyanarayanan 2001). Until recently, the availability and cost of devices – sensors – and networks have been practical barriers to the ubiquitous deployment and widespread acceptance of Context-Aware Computing.

Context-Aware Computing research has been driven by the development of devices, mainly in the form of various sensors of location, proximity and presence. The enticing device appears as important as enhanced functionality, which continues to drive development of higher resolution by means of numerous robust and easily deployed sensors. However, in seeking the goal of making everyday interactions with computers and communications devices unobtrusive and effectively smarter through context awareness, sufficient hardware devices as the means towards this end already exist.

The significant challenges in achieving this lie in making use of already existing devices, through the design of systems and software which enable ease of deployment, network and sensor scalability; through sensor fusion rather than through precision; and enabling wider acceptance through design for user needs by human factored interfaces and increased human trust in the systems' care for privacy and security.

IROS, Aura and DiCPA architectures are different to some degree. IROS is designed to move data between devices, remote access and to coordinate these devices to work together. IROS uses a centralized architecture (Ponnekanti, Johanson et al. 2003) for programming abstraction, maintainability/deployment and a fast-recovery strategy. Aura is designed to minimise distraction when the users access ubiquitous computing applications. DiCPA is designed to provide the interactive and interaction process between user and the environment in their own domain or in an unfamiliar domain to recoqnise user location, user mobility and user activity and how the environment responds to user situation. DiCPA design is a distributed architecture; the recognition between familiar domains and unfamiliar domains is resolved using triangle resolution server (See Figure 8.1).

IROS is an Interactive Room (iRoom) Operating System. IROS can be defined as a ubiquitous computing environment where groups come together to collaborate on solving problems (not a toolkit). IROS contains *embedded devices*; large display screens, table-top display, and other output devices, and *mobile devices*; mobile devices brought into the space can be used with embedded facilities (Johanson, Fox et al. 2002, Johanson and Fox 2001).

The Aura Project builds an architecture framework for User Mobility in Ubiquitous; Computing by allowing users to move computational tasks from one place to another, maximize the use of available resources, minimize the user distraction and drains on attention. Personal information in Aura spans wearable, handheld, desktop, and infrastructure devices (Sousa and Garlan 2002).

An Intelligent Environment as a Ubiquitous Computing environment is an environment which has significant processing done by various fixed and mobile sensors. The Intelligent Environment in this thesis was built using DiCPA architecture. The DiCPA architecture is based on Merino services layer architecture as explained in (Kummerfeld, Quigley et al. 2003). The DiCPA architecture was designed to be a scalable context-processing architecture, which contains at least an Intelligent Environment repository, a resources manager, a knowledge base and a global/local resolution object naming scheme. It manages the interaction between users and the environment in the Intelligent Environment domain by handling raw sensor information and making transformations between the raw physical devices/sensors within the environment and the application programs. The infrastructure supports interoperability of context sensors/widgets and applications on heterogeneous platforms.

An Intelligent Environment infrastructure can be implemented by a wireless data communication network such as a WVLAN combined with a General Packet Radio Service (GPRS) or a Universal Mobile Telecommunications System (UMTS). The WiFi (802.11b,b+,g), Bluetooth, IrDA and GPRS (GSM network) were implemented to integrate them with a local wired network. A MAC address of WiFi and Bluetooth were used to identify users, with these technologies also making it possible to determine user location in an indoor environment (Mantoro and Johnson 2003; Mantoro and Johnson 2003; Mantoro and Johnson 2004).

The term "Active Office" is used as an implementation model of an Intelligent Environment. An Active Office can be defined as a normal office, which consists of several normal rooms with minimal additional decoration (minimal intrusive detectors and sensors, without explicitly badging people) and using DiCPA architecture to manage and respond to rapidly changing aggregated sensor data. For the Active Office an Intelligent Environment is the collection of detectors and sensors in a local physical vicinity which corresponds to a "local domain" of the Intelligent Environment, with descriptions and communication for these entities regarded as services. In order for an Active Office to provide services to the users, the Active Office can be designed to detect the user's current state/context and to determine what actions should be taken based on that context.

This chapter also describes the requirement for recognition of user locations and activities in an Active Office environment. First, an overview of the Merino architecture is presented which is followed by descriptions of the DiCPA distributed context processing architecture, Intelligent Environment domain, Intelligent Environment resolution, Intelligent Environment repository, user modeling, resources manager, knowledge base, and user's activities. The user's activities describe how a user model is characterised by user identification and authentication, user profile, network access and service adaptation to user environment. There is discussion of a scenario which recognises the user's location and activities. A summary will be presented in the final part of this chapter.

3.2 Merino Service Layer Architecture

The Merino architecture has two important parts (Figure 3.1). The first part is the management of the interaction between physical devices/sensors within the Intelligent Environment and the program application, the management of the interaction between the user and the environment. This part contains its abstraction layers: Core Sensors and Device Layers, Device Abstraction Layers, Context and the highest abstraction level, i.e., Smart Environment Agent Layers (Kummerfeld, Quigley, et al. 2003).

The Sensor Layer is the innermost service layer and represents the range of sensors which detects the physical environment. The next layer is the Context Layer, which performs the core task of filtering and aggregating the raw data from the Sensor Layer and also ensures interoperability between sensors.

The second part consists of an Intelligent Environment Repository and a User Model. An Intelligent Environment Repository is a key element that unifies and manages the whole collection of objects in the environment. The Context Layer interacts with the Intelligent Environment Repository. This handles a global name structure for objects and manages the naming/subnaming authority.

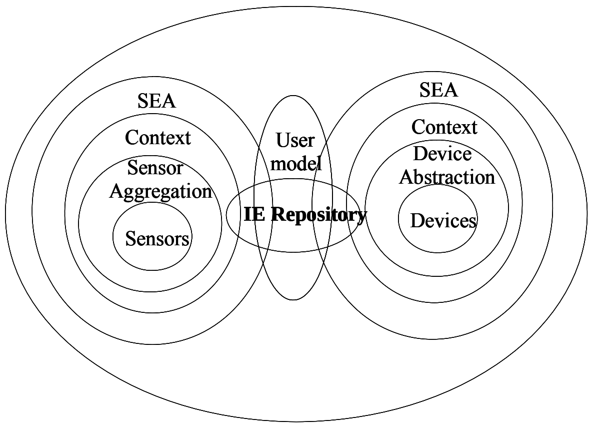


Figure 3.1 Merino Service Layer Architecture for the Intelligent Environment

A User Model overlaps with the Intelligent Environment Repository. The Merino architecture treats them almost the same; the main difference is that the User Model concerns people data only. The User Model contains personal identity, which is subjected to security and privacy to conform to the emerging environment, in a local and a global space. On the other hand, the Intelligent Environment Repository holds context information from the Context Layer. Data from the Intelligent Environment Repository which is associated with the user will be stored in the User Model.

For example, data from a movement detector, a WiFi or a Bluetooth proximity detector, and a keyboard activity monitor will be held in the Intelligent Environment Repository, but once it is associated with an individual user, it will be kept in the User Model. The device and the Device Abstraction Layer are motivated by the need to send data to low level devices in the Intelligent Environment using some device attribute, e.g. a phone number or a Bluetooth MAC address in a mobile phone. A Smart Environment Agent may determine that the context implies that "a meeting is in progress", and communicates with the phone to change its state, e.g. to request switching to a silent mode.

3.3 DiPCA: Distributed Context Processing Architecture

An Intelligent Environment is a computing environment (indoor or outdoor space) with significant processing done by various fixed and mobile sensors, which contains at least an Intelligent Environment domain and a global/local resolution object naming scheme. An Intelligent Environment is also a computing environment with capability to recognise user location, activities, and the social context defined by the presence of other people. As an implementation model of an Intelligent Environment, Active Office is developed as an office equipped with various (wireline and wireless) sensors networks and using scalable context processing architecture.

DiCPA is a mobile distributed context-aware processing architecture for an Intelligent Environment. In this architecture, an Intelligent Environment domain plays an important role in distributed context processing that are managing and administering one or more domains. In an Intelligent Environment domain, a resources manager is a network management map application that provides information about the status of an object (device/sensor) in the Intelligent Environment network. The resources manager detects, controls, manages and concludes the functionality of all objects in the Intelligent Environment domain (Mantoro and Johnson 2004).

Context layer architecture is designed as the basis for and an integral part of DiCPA architecture to enable the system to recognise user activity. Four layers are introduced, i.e., Context Application Layer, Context-based Layer, Abstraction Layer and Transport Layer (Figure 3.2). For more technical detail as to how a scalable distribution context processing architecture was designed please refer to (Mantoro and Johnson 2004).

The Context Application Layer is a layer that interacts directly with a variety of user activities, therefore it needs to address several issues including "security", "access right" and "privacy rules". In this layer "security" is considered to be who has access to the information and "access right" is considered to be the degree of accuracy/detail which may be accessed and "privacy" is considered to be to what extent information may be used.

This layer also considers a flexible and scalable interaction mechanism for interactions between user and the Intelligent Environment. Different clients can have different requirements for the mode of interaction (pull/push) and granularity of location information, for example (Indulska and Sutton 2003).

The Context-based Layer is a layer that contains many functions such as managing Intelligent Environment resources, providing resolution between Intelligent Environment domains and resolution servers, and managing the knowledge base. The knowledge base in this layer has the capability to merge with other Intelligent Environment domains and its growth could be controlable. This layer is responsible for shrinking, extracting and filtering the rules to produce better performance without losing their essential integrity (Mantoro and Johnson 2004). This layer has the capability of retrieving history data from the Intelligent Environment repository and of building relationships which employ user models to form user mobility patterns. It can be used for the prediction of user location (Indulska and Sutton 2003).

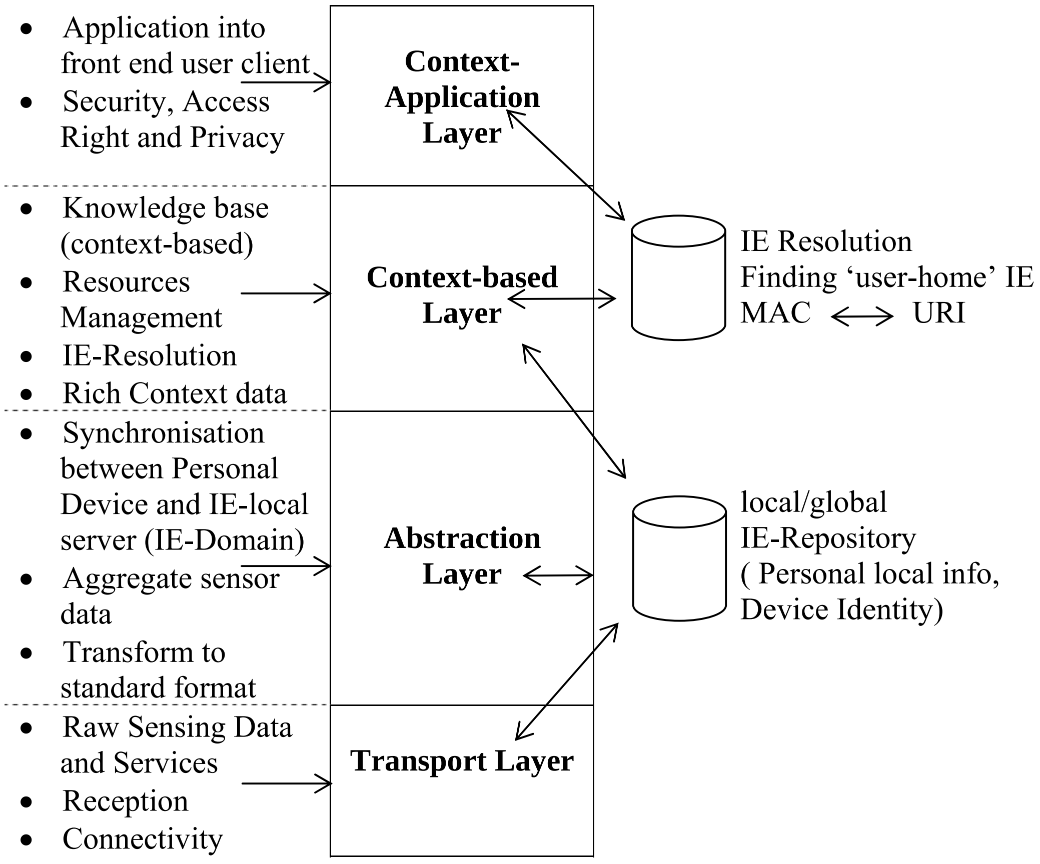

Application into front end user client
Security, Access Right and Privacy
Context-Application Layer
Knowledge base (context-based)
Resources Management
IE-Resolution
Rich Context data
Context-based Layer
IE Resolution
Finding 'user-home' IE
MAC ⟷ URI
Synchronisation between Personal Device and IE-local server (IE-Domain)
Aggregate sensor data
Transform to standard format
Abstraction Layer
local/global IE-Repository
(Personal local info, Device Identity)
Raw Sensing Data and Services
Reception
Connectivity
Transport Layer

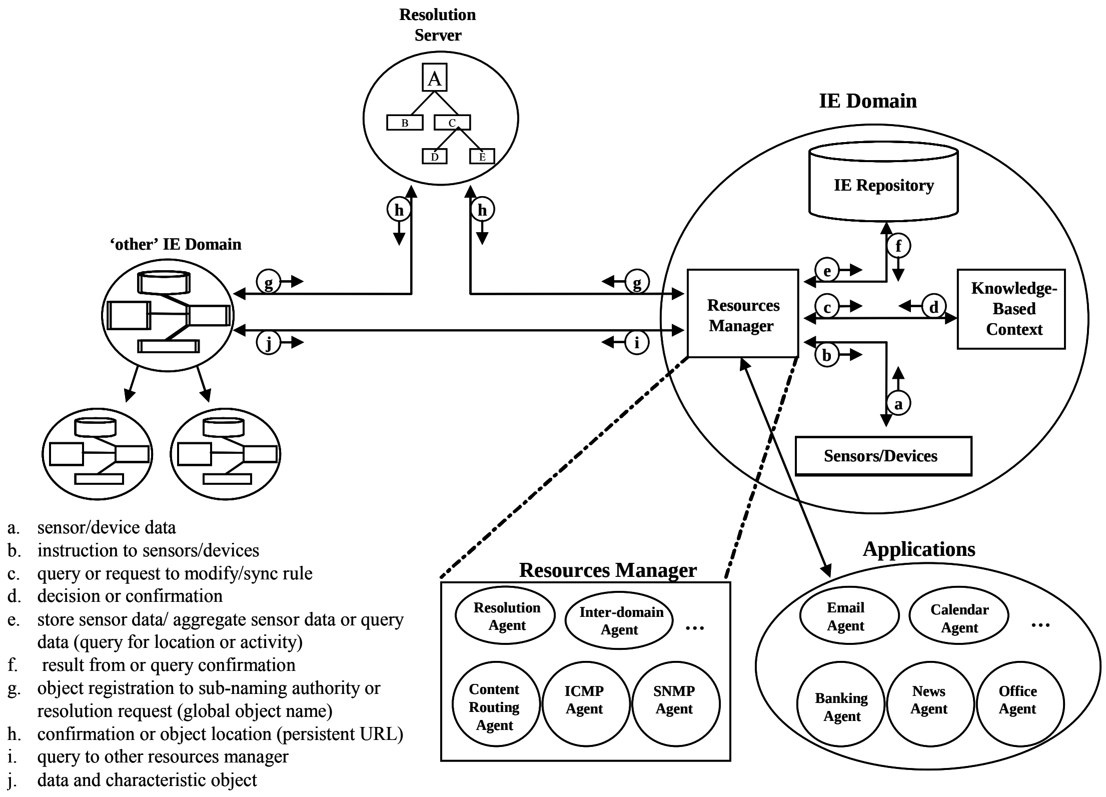
Figure 3.2 Context Layer Architecture

The Abstraction Layer is a layer which deals with the transformation of raw sensor data from the Intelligent Environment Repository to a standard format, and then aggregates the formatted sensor data. The data can be synchronised between the personal user device and the Intelligent Environment local server in the Intelligent Environment domain. This layer also manages the relationship between different places in Intelligent Environment domains.

The Transport Layer is a layer which manages reception and connectivity between a sensing device and the user device. It has the capability to accept a new type of sensor easily. A physical sensor agent collects raw sensing data stored in a local Intelligent Environment repository (e.g. WiFi signal strengths and Mac Address of the device).

3.3.1 Intelligent Environments Domain

DiCPA is defined as a scalable context processing architecture with the capability of delivering service while the user is moving about in an Intelligent Environment. DiCPA architecture uses an Intelligent Environment domain as a key role in communication between domains. More over, an Intelligent Environment domain is also an administrative domain, which contains at least an Intelligent Environment repository, a resources manager, a knowledge base and various sensors (above the level of dumb sensor that communicates by using a standard protocol).



a. sensor/device data
b. instruction to sensors/devices
c. query or request to modify/sync rule
d. decision or confirmation
e. store sensor data/ aggregate sensor data or query data (query for location or activity)
f. result from or query confirmation
g. object registration to sub-naming authority or resolution request (global object name)
h. confirmation or object location (persistent URL)
i. query to other resources manager
j. data and characteristic object

Figure 3.3 DiCPA: Distributed Context Processing Architecture for an Intelligent Environment

A resources manager uses sensor/widget agents to have direct communication between sensors and devices to the Intelligent Environment repository. A resources manager also distributes the sensor data or the aggregated sensor data to other Intelligent Environment repositories in other Intelligent Environment domains. The structure of an Intelligent Environment domain can be hierarchical (it can have a single higher level Intelligent Environment domain or have several lower level Intelligent Environment domains) or be scattered without any level (Figure 3.3). In DiCPA architecture, the communication between Intelligent Environment domains is based on a request from each resource manager (peer-to-peer basis). The resources manager will send a request to the resolution server to get all information about an object. The resolution server accepts the object's global name and uses persistent mapping from the object's global name to send the persistent location (persistent URL) of the object to the resources manager. Finally, the resources manager sends a request to a relevant resources manager in another Intelligent Environment domain to get the object.

Every object in the Intelligent Environment domain is defined as being a self-describing data-structure. This means that the structures as well as the value are always included in the object's content. This approach has two advantages: first, it avoids the complexity of separate schemes and the maintenance of objects across multiple distributed servers, and second, it permits meta-tools, e.g. a browser, to manipulate the object without knowledge of specific content (Schilit 1995). A particular object has a unique identifier and is located using a search that starts in the local Intelligent Environment domain, and expands to an adjacent and higher level Intelligent Environment domain such as a resolution server (using persistent URL or Handle system).

Communication within an Intelligent Environment domain uses logical multicasting. This means that communication between an Intelligent Environment repository and sensors could be implemented using IP multicast with well known group-content-filtering in the receivers or message server distributed content-routing and filtering (such as Elvin or Spread) for an automatic update of object content (Segall, Arnold et al. 2000).

For example, consider a person moving between locations. Anna works in the Computer Science Department. The CS department (IE) domain contains the home server for her personal user model. When she walks to the nearby Department of Engineering building, the CS department context layer will use aggregated sensor data to determine that Anna is not present, and other aggregating sensors in the Department of Engineering will detect her presence as an unknown object and ask their local resources manager for the address of the server for this object. The local resources manager multicasts the request to the resolution server and discovers the correct home server for the object – her user model. The object's location information in the home server is then updated accordingly.

3.3.1.1 Intelligent Environments Repository

The Intelligent Environment's repository keeps all objects and their characteristics. It holds the rapidly changing state of the objects. It contains all relevant data within the Intelligent Environment domain, e.g. sensor data, aggregated sensor data, context data and rich context data. The Intelligent Environment repository uses a distributed object database for Context Repository and User Model purposes. It also contains user profiles, registered devices (server, workstation, bridge, router, switch, printer, phone, fax), sensors, network (type - wired or wireless, subnetting, segmentation), data management, location mapping (sensors/devices signal-strength mapping to physical location), and other services data (monitoring data (SNMP/MRTG/ntop), hosts).

Every object in the Intelligent Environment repository has an identifier which is registered in the resolution server for Intelligent Environment resolution purposes, e.g. from URN to URLs.

The resources manager manages communication between Intelligent Environment repositories and sensors/devices using logical multicasting, with communication between

Intelligent Environment repositories using peer-to-peer (P2P) systems. These P2P systems use a distributed hash table (DHT) functionality and exact-match query facility. These systems are extremely scalable; lookups can be resolved in log *n* overlay routing hops for an overlay network of size *n* hosts. Exact match lookups are suitable for fetching objects or resolving domain names (Harren, Hellenstein et al. 2002). To do the lookups, extended SQL can be used.

DHTs are used not only as an indexing mechanism for all objects in the Intelligent Environment domain, but also as a network routing mechanism.

3.3.1.2 Intelligent Environments Resolution

In the Intelligent Environment domain, all objects can be distributed to another physical location or logical location. Every object has a global resolution name and the global resolution name needs to be registered in the resolution name server, e.g. PURL or Handle Server for persistent global mapping/resolution purposes. The resolution mechanism is identical to the DNS for the Naming Authority Pointer (NAPTR) Resources Record (RR) to delegate the lookup's name (Daniel and Mealling 1997; Mealling and Daniel 2000). The Unique Resolution Name (URN) is used as a global unique name and the Unique Resolution Locator (URL) as a locator which locates any object, anywhere, at anytime.

The purpose of Intelligent Environment resolution is to determine object identity when the object or the user changes its location, especially for recording dynamic location due to user mobility in an unfamiliar Active Office.

3.3.1.3 Resources Manager

A resources manager is a network management map application that provides information about the status of the object (devices/sensors) in the Intelligent Environment network. The resources manager detects, controls, manages and concludes the functionality of all objects in the Intelligent Environment domain.

The resources manager will show any failures in the devices/sensors, access points, hubs, bridges and routers. Furthermore, the resources manager will detect any traffic problems and will identify what, where, cause and time-length of problems.

The resources manager has the capability of controlling and managing the functionality of an IP network using the Internet Control Message Protocol (ICMP) and a 'trap' mechanism using the Simple Network Management Protocol (SNMP), so that a failsafe mechanism procedure can be implemented. To set up the resources manager, a set policy (knowledge-based context), that determines acceptable levels of traffic, and which broadcasts errors on any devices/sensors at any segments, needs to be established.

The resources manager knows all existing objects in the Intelligent Environment domain by querying the Intelligent Environment repository (context repository or user model layer). If the object does not belong to its Intelligent Environment domain, the resources manager has the mechanism to find the object by querying the resolution server to get the 'home server' and continue querying the 'home server' about the details and characteristics of the object if it exists. If it does not exist, the resources manager will tag it as an unidentified object.

3.3.1.4 Resources Manager Applications

Resources manager applications consist of user agents which include email agent, calendar agent, banking agent, news agent and office agent. The application will be in the users' handheld computer and will communicate to the Active Office through the resources manager. As shown in Figure 3.2, the application lies outside the Intelligent Environment domain. It makes it easier for the user to move from one room to another, even to another and unfamiliar Intelligent Environment domain.

The resources manager manages service delivery to the user. All information in the user model can be retrieved from the Intelligent Environment repository and be stored directly in the user's application by the resources manager and applications agent. The application allows the user to use available features in the Active Office based on the users' role. The user could also make an enquiry by using the context user interface.

3.3.1.5 Knowledge-Based Context

A complex infrastructure and service environment requires more shared knowledge for the entities to be able to invoke communication sessions. The knowledge base service must lead to the growth of services in an Intelligent Environment. The entities require a knowledge-based context, i.e. an advanced knowledge base founded on context service and a shared knowledge base. The knowledge-based context requires a protocol, e.g. JINI, JXTA, and UPnP, to invoke the services.

Entities are able to support a knowledge-based context which represents the capability of the entities and associated objects to create relations between them. Entities can exchange knowledge bases, merge with existing knowledge bases and then interpret the knowledge-based context at the end of the services.

When knowledge bases are merged, the new knowledge base will have the potential to grow considerably, then the knowledge base will need to shrink, extract and filter to achieve better performance but without losing the essence. Multiple Classification Ripple Down Rules (MCRDR) (Dazeley and Kang 2004) and Self Organizing Map (SOM) of Artificial Neural Network (ANN) components can be the best candidates to filter 'rules' to produce knowledge maps while capable of extracting raw and semantic rule features from sensors/devices.

The knowledge-based context contains context rules for interaction between the user and the Active Office environment. Rules are used to represent heuristics which specify a set of actions that need to be performed in a given situation.

A generic rule is composed of an *if* portion and a *then* portion. The *if* portion of a rule is a series of patterns which specify the facts from sensors in the Intelligent Environment where the rule would be applicable in the Intelligent Environment domain. The process of matching the facts to the patterns (pattern-matching) is done by an inference engine agent, a specific agent in the resources manager. The inference engine agent will automatically match the facts against the patterns and determine which rules are applicable to the context.

The *then* portion is a set of actions that needs to be executed when the rule is applicable to the situation (context). The actions of applicable rules are executed when the inference engine agent is being instructed to begin the execution. The inference engine agent selects the rules, and then the actions of the selected rule are executed. The *if* portion detects current status and the *then* portion determines what actions are to be taken based on the context.

The knowledge-based context also has dynamic knowledge cooperation that allows the rules and the sensor/devices to dynamically affect the actions-service selection process within the Intelligent Environment domain or the inter Intelligent Environment domain. The mechanism of the knowledge-based context described above can be implemented by using a rule engine and a scripting language, e.g. a Java Expert System Shell (JESS), a Prolog (such as SICtus prolog), CLIPS, or other function languages.

3.3.2 Subject and Environment Role-Based Access Control

A very important aspect of an Intelligent Environment is providing users with security and privacy, which models access control with richer contextual information.

A variant of role-based access control (RBAC) was developed for the Intelligent Environment. Given the sensitivity of information that is generated and stored in such an environment, as well as the many complex interactions that will take place in the Intelligent Environment, security policies can be potentially quite complex and access control must take into account the richer contextual information.

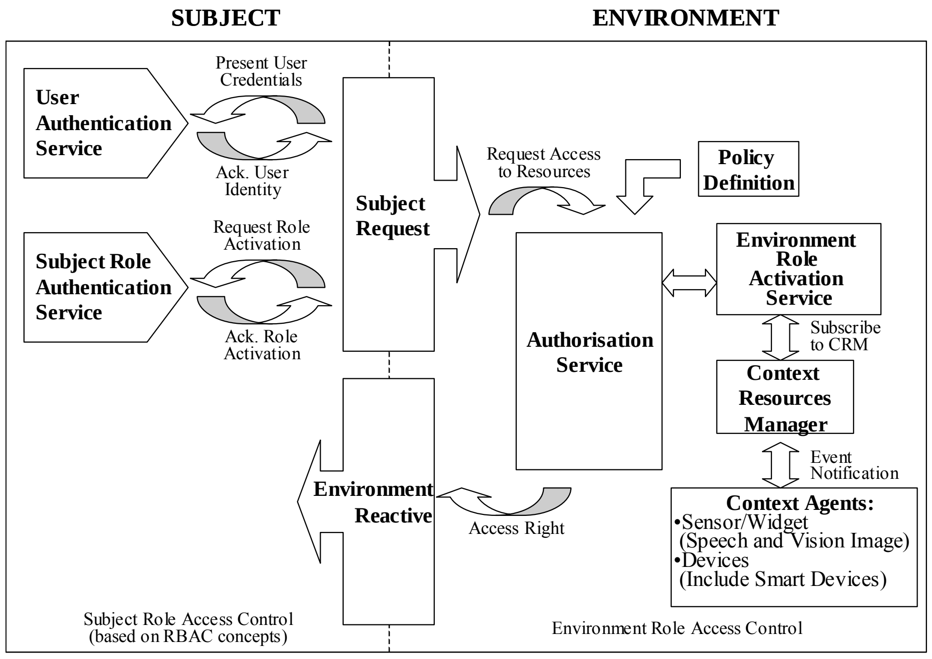

SUBJECT
ENVIRONMENT
User Authentication Service
Present User Credentials
Ack. User Identity
Subject Role Authentication Service
Request Role Activation
Ack. Role Activation
Subject Request
Request Access to Resources
Policy Definition
Environment Role Activation Service
Authorisation Service
Subscribe to CRM
Context Resources Manager
Event Notification
Context Agents:
•Sensor/Widget (Speech and Vision Image)
•Devices (Include Smart Devices)
Environment Reactive
Access Right
Subject Role Access Control (based on RBAC concepts)
Environment Role Access Control

Figure 3.4 Block Diagram of Role-based Transactions for a Distributed Intelligent Environment

Unfortunately the existing paradigm of traditional software engineering cannot be used as a basis for constructing an Intelligent Environment. It needs to be conceived as an information processing system where a largely invisible and Ubiquitous Computing infrastructure is integrated and different types of knowledge are properly represented to assist people with a variety of activities in mobile computing use. To do this, role-based transactions for distributed support for the Intelligent Environment need to be developed.

Traditional RBAC offers an elegant solution to the problem of managing complex access control-rule sets. The basis of RBAC is the concept of a role, where a role is a grouping mechanism that is used to categorise subjects based on various properties. Such properties include user functions or responsibilities.

Although RBAC is very useful for modeling access control in a variety of applications, its roles are inherently subject centric (Sandhu, Coyne et al. 1996; Covington, Long et al. 2001). It cannot be used to capture security-relevant contexts from the environment, which could have an impact on access decisions. In this work, the RBAC model is extended to allow policy designers to specify such environmental contexts through a new type of role, namely the *environment role*. It means that the RBAC model has two fundamental roles, a subject role and an environment role. Figure 3.4 shows the design of the RBAC model, which can be used and implemented to enable the user to have correct access to context-aware applications.

3.4 The Application Scenario

In this part, a scenario where users and devices create a connection in an unfamiliar Intelligent Environment domain is described.

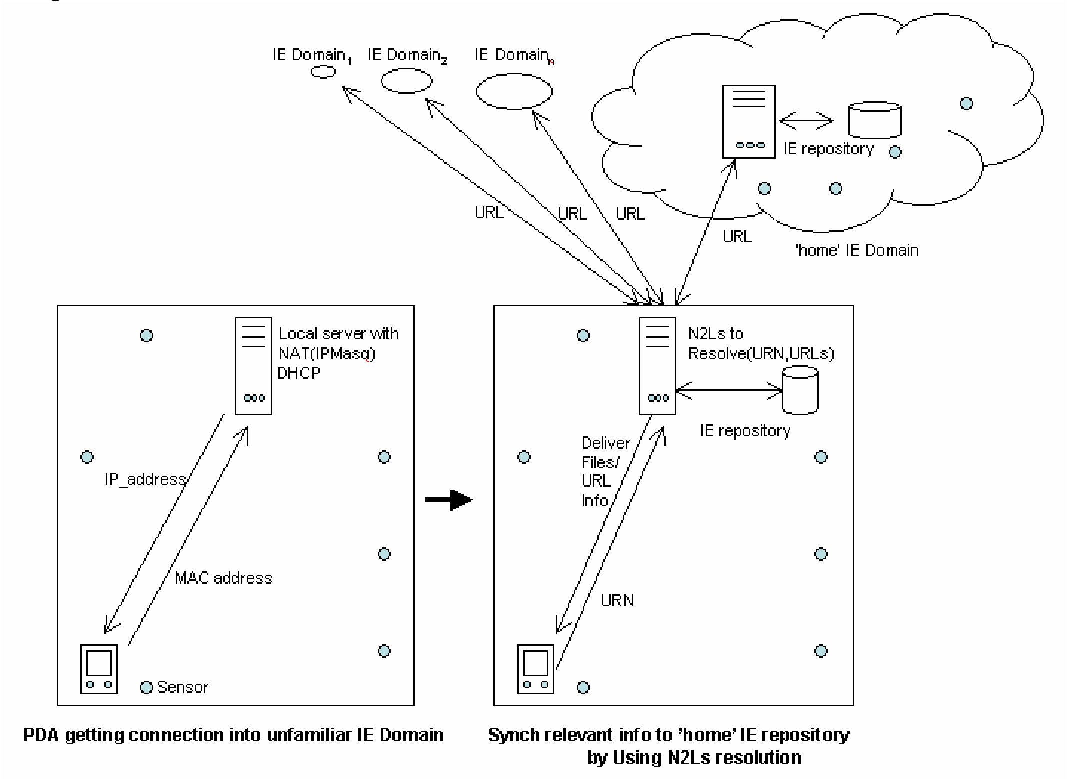

IE Domain$_1$
IE Domain$_2$
IE Domain$_n$
IE repository
URL
URL
URL
URL
'home' IE Domain
Local server with NAT(IPMasq) DHCP
IP_address
MAC address
Sensor
N2Ls to Resolve(URN,URLs)
IE repository
Deliver Files/ URL Info
URN
PDA getting connection into unfamiliar IE Domain
Synch relevant info to 'home' IE repository by Using N2Ls resolution

Figure 3.5 Making Connections with an Unfamiliar Intelligent Environment Domain

In the Active Office, assume that the user brings a portable computer, i.e. a mobile host (smart PDA), rather than a simple personal assistant or a pager or phone. The user's mobile host has reasonable capabilities in processing power and memory/storage, remote access and an execution mechanism such as Remote Procedure Call (RPC) so that a fixed local server could run any applications which are unavailable on the mobile host. A user, for example, wants to deliver a presentation from a file at a remote server using facilities in an unfamiliar Active Office with an Intelligent Environment domain.

How this unfamiliar Active Office responds to the user's presence and requests will be described. An unfamiliar Active Office can be assumed to have a fixed server through a wired LAN which connects to the Internet and the server having a reasonable service to connect to the mobile host through a wireless connection, e.g. IEEE 802.11b, 802.11g, Bluetooth or IrDA, as part of the network infrastructure.

First, the user logs in to his PDA. When the user arrives at an unfamiliar Active Office, his PDA will broadcast its unique hardware identifier, i.e. vendor-supplied MAC or BDA address. The local server has a non-authoritative DHCP and a Network Address Translation (NAT) service. The NAT uses IP Masquerading to translate IP addresses. After a handshaking process, the server will send a local IP address to the PDA. At this stage the PDA has established a connection to the network (Figure 3.5).

There are several ways to get connected to the home server, e.g. using NIS(YP), NIS+, or mounting a directory to the home server. However, these ways are not appropriate for Active Office purposes since it has only limited information, i.e. a MAC address, instead of home server, login ID and password which are required for use of NIS+ or a mounting directory. Therefore the resolution server is used to get connected to the 'home server'. When the Intelligent Environment domain identifies an unknown object (the MAC address of the PDA), its resources manager will send this MAC address in a query to the resolution server to retrieve a URN.

For example (based on RFC 2168 an Updated RFC 2168):

$$\begin{tikzcd}[column sep=small] \textit{http://} \arrow[d] & \textit{resolution.anu.edu.au/} \arrow[d] & \textbf{\textit{1030.52}} \arrow[d] & \textit{/00-80-BD-08-08-08} \arrow[d] \\ \text{protocol} & \text{resolution-server} & \textbf{subnaming-authority} & \text{MAC-address/BDA} \end{tikzcd}$$

The response from the resolution server, for example is *jblog@inul.org.*

The resources manager will send another query using this URN to get URL resolution.

For example:
GET /usi-res/N2L?urn:1030.52:jblog@inul.org HTTP/1.0

The result would be:
# urn: 1030.52:john.blog@inul.org
http://www.inul.org/people/john.blog.html
http://www.inul.org/pub/presentation.pdf
ftp://ftp.inul.org/pub/presentation.ppt

The resources manager in the unfamiliar Active Office then will get information, such as the person's name and administrative domain and where the file for the presentation is available. When the resources manager stores the location information to the user’s PDA, at the same time the resources manager in the ‘home’ domain changes the location status in the Intelligent Environment repository (user model layer) (Mantoro and Johnson 2003; Mantoro and Johnson 2003).

The user’s location has been implemented based on history data (predicted user location) and a simple user interface using speech recognition called SpeechCA developed using Java Speech API (Mantoro and Johnson 2003). Further details of SpeechCA are given in Chapter 4. Service delivery, based on user location and user’s mobility in an Active Office, has been studied where the user has moved to ‘another’ Intelligent Environment domain: he still has full access to resources including the internet, but for incoming and outgoing data he has to use his local Intelligent Environment domain using WiFi while he can print to the closest printer in ‘another’ Intelligent Environment domain using Bluetooth (Mantoro and Johnson 2003; Mantoro and Johnson 2003).

3.5 Summary

This chapter proposed DiCPA architecture - a distributed context processing architecture - for an Intelligent Environment which is simple, efficient, scalable, fault tolerant and applicable to be implemented across a range of heterogeneous computing platforms.

This chapter also described the requirement for scalable context processing in an Active Office using DiCPA architecture as an implementation model of an Intelligent Environment. An overview of the Merino service layer architecture and distributed context processing architecture has been presented followed by a description of Intelligent Environment domain, Intelligent Environment resolution, Intelligent Environment repository, user modelling, resources manager, knowledge base, application and interoperation between sensors.

The scalable distribution of the context is shown as well as how user model information is managed in the recognition of user activities. An illustration was provided to show how users get connected in an unfamiliar Active Office to another Intelligent Environment domain and how to monitor user activities.

Chapter 4

LOCATION AWARENESS IN INTELLIGENT ENVIRONMENTS

Location awareness is a crucial part of the context-awareness mechanism for Ubiquitous Computing. The problem is how the Intelligent Environment determines user location from a variety of sensors which have different precision and data types. This chapter describes a user location model, the model that uses various sensors to find, to detect and to predict user location in an Intelligent Environment. In an Intelligent Environment, user locations are categorised by three location types i.e. Precise Location, Proximate Location and Predicted Location. The Precise Location and Proximate Location categories are based on the sensor's capability in covering an area. Predicted Location is based on history sensor data from Precise Location and Proximate Location.

4.1 Introduction

Location-awareness is a very important aspect of context-awareness for mobile computing systems. The off-the-shelf availability and everyday use of a number of moderate-cost mobile devices (PDA, smart phone, handheld and laptop computers), installed wireless and wired networking, and associated location information, leads to a focus on Context-Aware Computing that rests lightly on everyday environments, and which asks in particular: "What effective location-aware computing can be achieved with minimal, unobtrusive, commodity hardware and software?"

The indoor office environment is a promising target for location-awareness and still presents research challenges. Many everyday tasks performed by people in the office are mediated by quite low-precision knowledge of other people's locations, such as whether the users are in the building, in an area of rooms, in a specific room, and so on. For example, deciding how to contact someone (by email, by phone, face to face); timing the issuing of a reminder for a scheduled event depending on the expected travel time of the event, given locations of person and event and some knowledge or model of the time needed; determining how to deliver an incoming message notification, depending on physical and social location of addresses. The human decisions associated with these traditional tasks can be assisted by the evolutionary step of providing people with information from location-aware systems, without taking the revolutionary and less acceptable step of their total replacement by computer-mediated communication.

The department-office environment at ANU with its existing installed network of numerous desktop and mobile computers can itself become our laboratory, in the usual research paradigm of Context-Aware Computing. With additional sensors and network hardware, this laboratory environment can demonstrate that software incorporating low levels of reasoning supported by lightweight knowledge is able to provide such helpful location-awareness services.

The experimental environment for an Active Office is deployed to provide the location-context awareness communication service application. The development of this application is mainly motivated by the need to demonstrate the effectiveness of communication between user and environment. It uses speech recognition technology. It gives assistance with existing everyday tasks, location and time elements of context in a multiple-room, multiple-linked building office environment available as a tool across currently used office-wide devices, desktops, semi-mobile laptops, and highly mobile PDAs (Smart PDA). The speech recognition application for location awareness service application is called SpeechCA.

The SpeechCA is a prototype of location context agent using speech recognition technology. This prototype is in the first stage of a system combining location sensing and deduction, and speech technology. The camera and voice input and sound output, releases users from keyboard, stylus and mouse and allows them to interact with their computational environment more in the way that they do with other people. However, in the physical Intelligent Environment model, any interaction has to have an object location and object identity. In this chapter the object location and the prototype of location awareness service is discussed and for the fusing of sensor data from various types of sensors is discussed in Chapter 8.

4.2 Location Context-awareness

Location is a very important aspect of context for mobile users, when finding the nearest resources, navigation, and locating objects and people. Location in the context-awareness application needs the modelling of the physical environment and the representation of the location (Harter and Hopper 1994; Schmidt, Beigl et al. 1999).

Numerous location models have been proposed in different domains, and can be categorised into two classes (Jiang and Steenkiste 2002):

- Hierarchical (topological, descriptive or symbolic, such as a room name).
- Cartesian (coordinate, metric or geometric, such as GPS).

The location representation in most context-aware applications adopts a distributed collaborative service framework that stores location modelling data in a centralised data repository (Strang and Linnhoff-Popien 2004). Location related queries issued by end-users and other services are handled by a dedicated location service. This distributed service paradigm is attractive because of its scalability and modularity (Jiang and Steenkiste 2002). However an effective and efficient location representation method is needed to make this work.

In an Intelligent Environment there may be thousands of known places, devices, sensors and services. For one location/place there maybe many way to refer. Any set of referents for location reference is called *location authority* (Shafer 2003). The location representation needs a single framework to manage the variation of location in how to refer and use locations. Location authorities in use today are so varied that it seems unlikely to define a universal data representative (Shafer 2003; Johnson, Carmichael et al. 2004). Shafer propose the use of a location-authority-independent representation of a location as a data object and a “universal federation” in which all location authorities combine to create a “virtual location authority” to achieve the world in one framework (Shafer 2003). The location authority can be described as the relative static properties and simple static relationships related to a spatial region in 1. a coordinate system, 2. a place naming system (symbolic/hierarchical location), 3. a landmark system, and 4.

displacement from landmark or named location, e.g. 19 meters north of N101.csit.anu.cbr.au (room no N101, CSIT building, The Australian National University, Canberra, Australia)

Both the Hierarchical and Cartesian location models can be used for representing locations and enabling the rapid changes of location information between distributed context services within a space.

The Hierarchical location model has a self-descriptive location representation. It decomposes the physical environment to different levels of precision. A tree structure to handle location structure is implemented and the location data is stored as an object/entity in a relational database model (Figure 4.1).

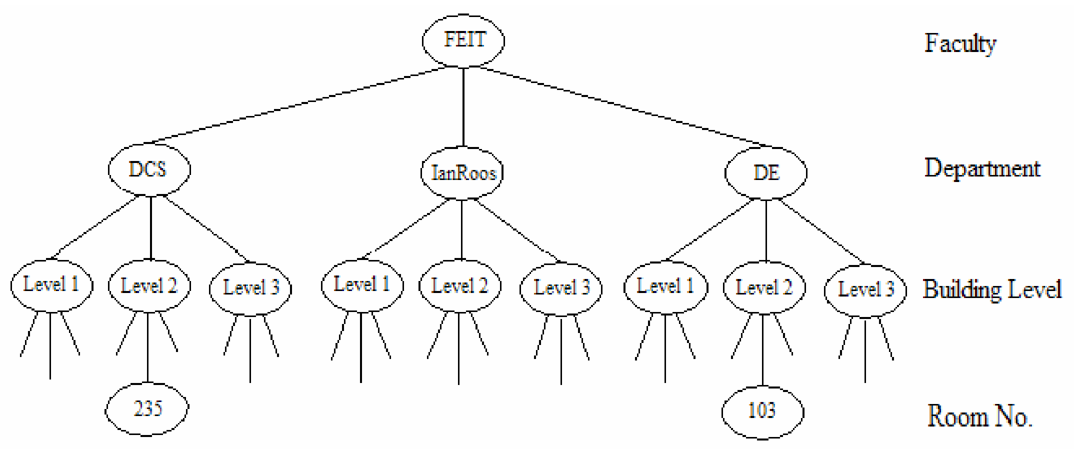


Figure 4.1 Example of Hierarchical Location Structure: Rooms in a Cluster of Buildings

An example of Hierarchical location is the reference location or landmark address from Table 4.1: Room database. The database uses a location-id rather than a room-no because in each of the three buildings, they have their own standard room names. The location id is needed to identify/tag the location properly in this system.

landmark_address=IE://Australia/Canberra/TheAustralianNationalUniversity/FEIT/DCS /2nd Floor/ Room 235.

Table 4.1 Example of Room Database

RoomNo	LocationID	Level	Building	…
235	125	2	DCS	…
103	323	2	DE	…
…	…	…	…	…
211	121	2	DCS	…
223	123	2	DCS	…
237	124	2	DCS	…
…	…	…	…	…

Cartesian location uses a grid on a physical environment and provides a coordinate system to represent locations. GPS uses a coordinate system that is defined by relative location in a space using the Cartesian system (longitude, latitude, altitude), e.g. Joe is located in (x, y, z). This chapter will not explore the shape and other extension location of the object.

Any simple mobile device has a physical location in space. At the spatial dimension, there are some devices (e.g., GPS-based map systems) where the Cartesian position is important in defining 2D or 3D space in a sense of absolute physical location. If location information is sufficient in understanding position, the location is considered in relation to other existing objects or sensors.

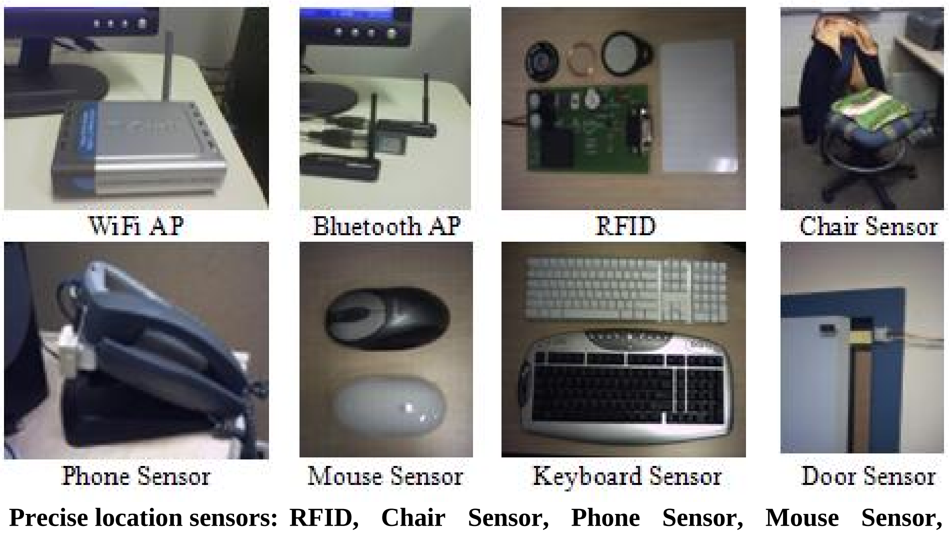
Unfortunately, GPS does not work precisely in an indoor Active Office environment, it has approximately 15 meters accuracy and its signal is too weak in an indoor environment. Newest GPS with WAAS (Wide Area Augmentation System) has approximately 3-5 meters (Garmin. 2005). GPS information is neither commonly available nor very accurate in the indoor environment. It works only for an outdoor space.

4.3 User Location Categories

In an Active Office, user location can be categorised as follows:

- Precise Location.
- Proximate Location.
- Predicted Location.

The Precise Location and Proximate Location sensors, as shown in Figure 4.2, are based on the sensor's capability in covering an area and Predicted Location is based on the history data of both Precise Location and Proximate Location.



Precise location sensors: RFID, Chair Sensor, Phone Sensor, Mouse Sensor, Keyboard Sensor, Door Sensor.
Proximate location sensors: WiFi AP, Bluetooth AP.

Figure 4.2 The Example of Sensors to Detect Precise Location and Proximate Location

The main problem in user location is how to combine these known location data to determine the user's actual location for office activity purposes; this is a matter of different precision and sometimes user location data is not available at arbitrary times because of hardware failure or on purpose by a user. How to characterise the pattern of user location based on aggregate current sensor data and history sensor data is also a problem.

The approach chosen is to aggregate the sensor data of Precise Location and Proximate Location and put priority on the determination of user location based on the precision of the sensors. When the Precise Location and Proximate Location are not available the history sensor data for Predicted Location is used.

4.3.1 Precise User Location

Precise location is user location based on sensors that cover less than a one-meter range, for example, swipe card, keyboard activity, mouse activity, biometric sensor/finger-print, iButton, RFID-tag, chair sensor, phone sensor, door sensor.

By registering the location of desktop computers, swipe cards, iButtons, fingerprints, RFID-tag, chair sensor, phone sensor, door sensor in an Active Office, the Active Office will have more precise user location information than from proximate sensors, i.e. WiFi or Bluetooth.

Fixed sensors such as a pressure chair sensor, a phone sensor and a keyboard activity sensor are very straight forward for user location. The information which the sensors capture is precise and correct. When a user sits on a chair with an embedded chair sensor in a certain room, the sensor data can be collected and stored to push to the spatio-temporal database. Unfortunately the pressure chair sensor in the Active Office implementation was not designed to recognise user identity. The Active Office needs another sensor to capture user identity from user profile such as a mobile phone with Bluetooth capability that is carried by the user.

4.3.2 Proximate User Location

Proximate location is user location based on a sensor that covers more than a one-meter range, for example, WiFi, Bluetooth, WiMedia, ZigBee, active/passive badge (depending on the range), voice recognition (microphone), face recognition (digicam), smart floor.

Proximate user location in an Active Office is detected by WiFi and Bluetooth sensors/devices. WiFi is an interesting proximate sensor because it can be used to access the network and also to sense user location within the scale of a room or an office. Bluetooth and IrDA (Infrared) can be used as a wireless personal area network. Both favour low cost and low power consumption, over range and peak speed. On the other hand, WiFi as a wireless local area network, favours higher speed and greater range but has higher cost and power consumption. The range of Bluetooth to sense another Bluetooth in a closed space, such as an Active Office, is about 3 meters for class 2 and 25 meters for class 1. The range of WiFi to sense a user with a WiFi device is about 25 meters (Mantoro and Johnson 2003).

Sometimes IrDA is not very useful in sensing a user's location because when there are many users in the area that is covered by IrDA, only one user will be recognised and only within 1 m range covering area. IrDA connections are limited to two devices at a time with direct line of sight (Hallberg and Nilsson 2002; Käppeli 2003). Bluetooth class

1 permits scanning about 25 meters (100 m in open space) between devices, Bluetooth signal strength can then be used for sensing user location. WiFi does not only have a higher speed and longer range than Bluetooth but the signal strength and signal quality of WiFi can also be used to detect user location in a room precision.

Bluetooth permits scanning between devices: when Bluetooth-capable devices come within range of one another, the location of one Bluetooth will be in the range of the other Bluetooth to allow scanning each other to form an ad-hoc network or other purpose. Bluetooth-capable devices can be used as sensors or an access point to sense a user with Bluetooth capable devices. Experiments unfortunately show that Bluetooth class 2's signal strength is not useful enough to sense a user's location. The signal is too weak. The Bluetooth can be used as an access point to sense several rooms within the range without measuring any signal strength. For example, when a user is close to a certain access point, the user's location will be proximately close to the access point. The result is not precise, because it could represent user location from several rooms.

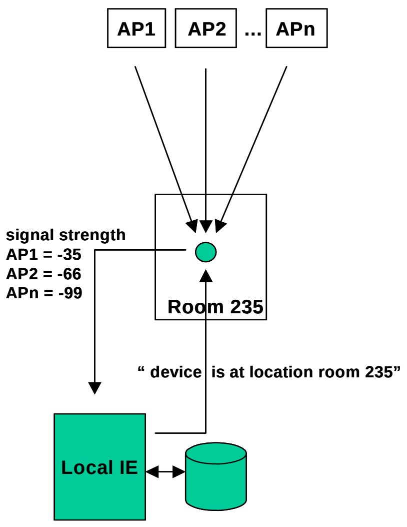


Figure 4.3 Device Measurement of WiFi APs' Signal Strengths

WiFi does not only have a higher speed and longer range than Bluetooth but the signal strength of WiFi can also be used to detect user location. Two scenarios are described as follows: First, the WiFi capable device determines the signal strength and signal quality. The signals data is stored to a local Intelligent Environment repository. The Intelligent Environment local server responds by sending the current user location. Second, the WiFi access point determines signal strength and signal quality. The signals data is stored in the local Intelligent Environment repository (Figure 4.3). The Intelligent Environment local server responds to a user device request by sending the user location. The difference between these scenarios is that in the first scenario the process of sensing is in the devices, with a user's mobile device with a high capability as the requirement, whereas in the second scenario the process of sensing is in the access point, hence it does not require a high quality of user's mobile device.

The problem in proximate location is combining the Bluetooth and WiFi known locations to determine the user's actual location for office activity purposes, this a different precision matter. Our approach has been to study signal behaviour in several rooms and corridors. The signal strength and signal quality of 802.11b/g is measured, collected and sent to an Intelligent Environment repository. A triangulation method cannot be used to estimate symbolic user location because WiFi signal strength and quality are easily changed, even by a very simple disturbance. The detail of the changing signal and also the cause was described in Section 5.5. The WiFi signal strength and signal quality need to cluster to estimate user location. A self organising map (Kohonen map) approach of an artificial neural network is used for these purposes.

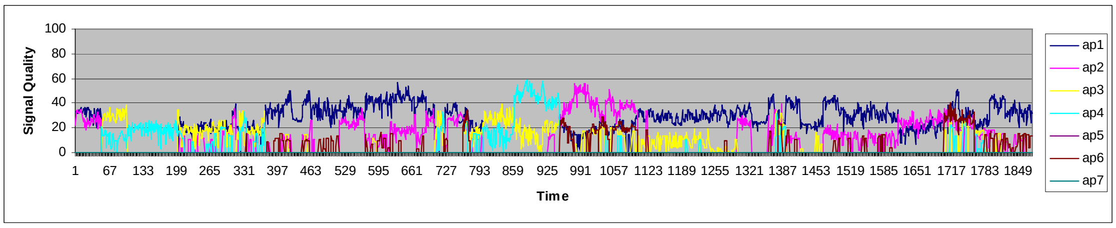
Kohonen has reported an interesting and useful result when a self organising map is used for pattern recognition tasks (Kohonen 1995). These maps classify a pattern represented by a vector of values in which each component of the vector corresponds to an element of the pattern. Kohonen's algorithm is based upon an unsupervised learning technique. Once trained, input data from a given class will produce excitation that represents the classification (Mantoro 1994).

Table 4.2 Example of Signal Strengths and Signal Qualities from Six WiFi Access Points

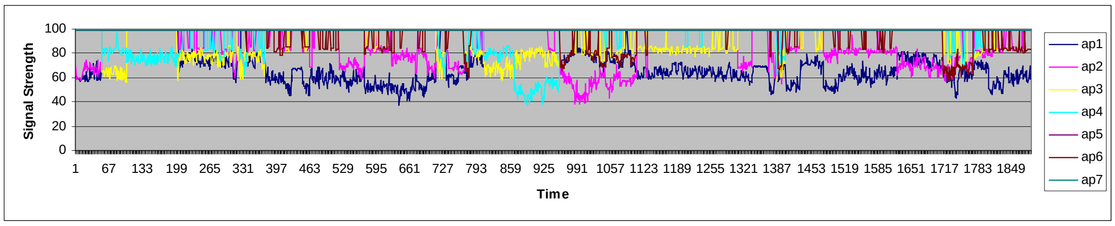
Signal Strength						Signal Quality						Room No
AP1	AP2	AP3	AP4	AP5	AP6	AP1	AP2	AP3	AP4	AP5	AP6	
…	…	…	…	…	…	…	…	…	…	…	…	…
58	66	99	99	99	99	36	28	0	0	0	0	N323
66	68	99	99	99	99	27	26	0	0	0	0	N323
99	99	99	73	99	99	0	0	0	21	0	0	DCScafe
99	99	65	78	99	99	0	26	26	14	0	0	DCScafe
…	…	…	…	…	…	…	…	…	…	…	…	…
99	99	57	82	99	99	0	0	38	4	0	0	DCScafe
99	99	56	84	99	99	0	0	37	7	0	0	DCScafe
99	99	99	73	99	99	0	0	0	19	0	0	Seminar
99	99	99	77	99	99	0	0	0	17	0	0	Seminar
…	…	…	…	…	…	…	…	…	…	…	…	…
99	99	99	75	99	99	0	0	20	20	0	0	Seminar
99	99	60	80	99	99	0	0	35	13	0	0	Wedge
71	83	76	81	99	99	23	12	19	15	0	0	Wedge
…	…	…	…	…	…	…	…	…	…	…	…	…

In this implementation, the WiFi's signal strength and signal quality are used to estimate user location from several access points. The building has 7 access points, but only the first 6 access points that provide the best signal strengths and signal qualities are used. The graphics on Figure 4.4 shows an example of the signals received in the case where a user changes his location within 7 hours (working hours for a day). It shows the variation of signal strengths and signal qualities from all 7 access points, from 0 dBm to 99 dBm on the y axes and time on the x axes. Table 4.2 shows examples of signal strengths and signal qualities from the 6 best access points.

A self organising map using a Kohonen map is suitable to cluster locations based on signal strength and signal quality data measurements in the local Intelligent Environment. The process to cluster signal strength data gives random weight to the artificial neural network followed by normalisation, calculation of Euclidian distance, and finding the winner for clustering. Training is continued until the learning rate is zero and the final weight is obtained, which leads to output activation and cluster-winner allocation to determine user location.



WiFi's Signal Strength of User Activity



WiFi's Signal Quality of User Activity

Figure 4.4 Example of Signal Activity from WiFi Sensors Within 7 Hours

In an experiment using WiFi, 13 access points are set up in two buildings to measure signal strength, 7 in the Department of Engineering and 6 in the Department of Computer Science. The detail of the building size and the spacing is described in Section 6.3. The results were good enough to estimate current user location in the Active Office area. On the 2nd level of the Department of Computer Science, most places had a good signal from more than two access points and could achieve accurate estimates (96%) in rooms of 3 meters width. On the 3rd level, where not all rooms/locations were covered by more than one access point, it could obtain only a reasonable degree of accuracy (75%) in predicting a user's current location (Mantoro and Johnson 2003).

The implementation of a self organising map strategy above is based on a cluster algorithm to cluster signal strength and signal quality, in the next chapter (Chapter 5) the signal strength and signal quality is grouped by room number, so clustering is not necessary anymore. To estimate user location, an instance-based learning algorithm using ηk-Nearest Neighbour is developed which eliminates clustering. This approach is simpler but reliable (see Chapter 5).

4.3.3 Predicted User Location (Location Context-aware History)

In an Intelligent Environment, ubiquitous sensors and actuators, simple push buttons and sliders, and computer access will be available in every area. Users can be identified by mobile computing devices (PDA/handheld), vision image recognition, active/passive badge, or by the activity of accessing available resources at physical static locations.

When user location is not available from Precise Location and Proximate Location sensors, the user's location can be identified by creating a history database of events, whenever the user accesses to identify himself (such as when using iButton or login to the network) or whenever the receptor/sensor/actuator (such as webcam, handheld, active/passive badge) captures the user's identity in a certain location. The following

(Table 4.2) is an example of a location history database for a user. Based on this historical data, the regularity and pattern of accuracy of user mobility and user activity such an in Chapters 6 and 7 can be analysed.

Table 4.3 Example of Location History Database

UserID	LocationID	Date	Time	Device
Joe	125	13/8/02	04.02	IButton
Joe	323	20/8/02	05.01	PDA
…	…	…	…	…
Joe	125	26/8/02	03.02	Sun15
Joe	125	26/8/02	03.58	Ibutton
Joe	125	27/8/02	04.05	PC5
…	…	…	…	…

History data from precise and proximate users' locations is collected and stored in the Intelligent Environment repository (see Sections 3.4.1 and 3.4.2). The pattern of user location based on aggregated history sensor data can be used to characterise predicted user location.

Table 4.3 is an example of location history data in an Intelligent Environment Repository, with history data above used to predict user location. Based on this data, a probabilistic model to find the most probable location of a user can be developed. History data also can be used to set policy in the log of history data to predict user location.

Aggregate precise location has first priority in determining user location and is followed by proximate and predicted locations respectively. This means that when the Active Office receives a query of user location, aggregate precise location will be used to determine user location, with the current user's location then determined. If the user precise location is not available, the aggregate proximate location data will then be used.

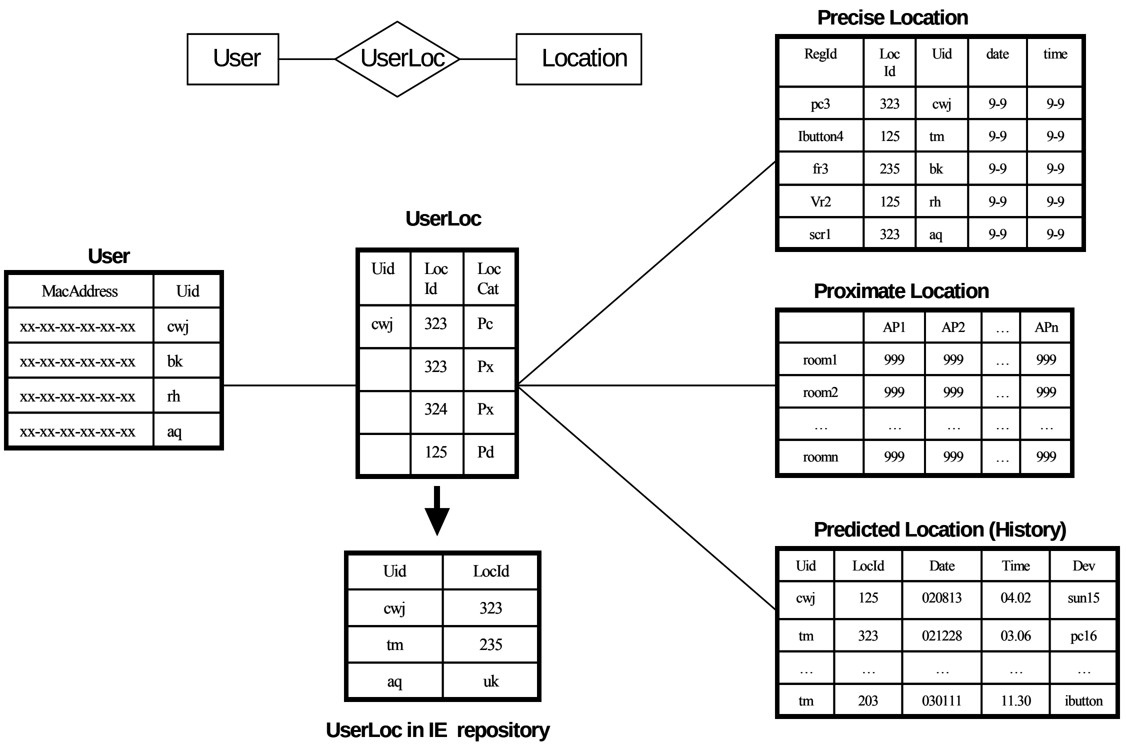
In the case of there being no user location in aggregate Precise Location or Proximate Location then Predicted Location will be used, as in the case, for example, where Joe works in room E213 which is not covered by WiFi or Bluetooth access points, and there is only a workstation present. If at the time of query Joe is not accessing the workstation, the history data is used to find, which is the Joe's most probable location based on history data for the same day at the certain time of the week or using all the days at the certain time in a one week range.

Experiments using Wireless LAN, i.e. WiFi and Bluetooth as proximate location, gave significant results in sensing a user's proximate location.

4.4 User Location Aggregation

This section discusses how the Active Office determines user location based on three location categories: precise location, proximate location, and predicted location. Figure 4.5 shows that an Active Office processes the information to determine a user's location by putting the priority to the user's location data or aggregating the user's location data from various sensor's data.

Aggregate precise location has first priority in determining user location and is followed by proximate and predicted locations respectively. This means that when the Active Office receives a query of user location, aggregate precise location will be used to determine user location, with the current user's location then determined. If the user precise location is not available, the aggregate proximate location data will then be used.



User

MacAddress	Uid
xx-xx-xx-xx-xx-xx	cwj
xx-xx-xx-xx-xx-xx	bk
xx-xx-xx-xx-xx-xx	rh
xx-xx-xx-xx-xx-xx	aq

UserLoc

Uid	Loc Id	Loc Cat
cwj	323	Pc
	323	Px
	324	Px
	125	Pd

Uid	LocId
cwj	323
tm	235
aq	uk

UserLoc in IE repository

Precise Location

RegId	Loc Id	Uid	date	time
pc3	323	cwj	9-9	9-9
Ibutton4	125	tm	9-9	9-9
fr3	235	bk	9-9	9-9
Vr2	125	rh	9-9	9-9
scr1	323	aq	9-9	9-9

Proximate Location

	AP1	AP2	…	APn
room1	999	999	…	999
room2	999	999	…	999
…	…	…	…	…
roomn	999	999	…	999

Predicted Location (History)

Uid	LocId	Date	Time	Dev
cwj	125	020813	04.02	sun15
tm	323	021228	03.06	pc16
…	…	…	…	…
tm	203	030111	11.30	ibutton

Figure 4.5 Aggregate Users' Locations in an Active Office

In the case of there being no user location in aggregate Precise Location or Proximate Location then Predicted Location will be used, as in the case, for example, where Joe works in room E213 which is not covered by WiFi or Bluetooth access points, and there is only a workstation present. If at the time of query Joe is not accessing the workstation, the history data is used to find, which is the Joe's most probable location based on history data for the same day at the certain time of the week or using all the days at the certain time in a one week range.

Experiments using Wireless LAN, i.e. WiFi and Bluetooth as proximate location, gave significant results in sensing a user's proximate location.

4.5 The Prototype of Location Context Agents using Speech Recognition

As generally known, for communication purposes, humans accept "input" information from eyes and/or ears, and have a special output "device" which is the mouth. The Intelligent Environment mimics this by having vision cameras for eyes and microphone for ears and the monitors and speakers are used as the output devices for communication between users and environments. In the Intelligent Environment, the user interfaces are not only pop-up menus, keyboards or mice but also gesture, speech, and movement (Coen 1998). It will communicate through context resources management for the availability of communications and computational resources in the Intelligent Environment. This will incorporate computers' capability into the real world by embedding the computer technology in regular environments and allowing people to interact with their computational environments in the same way they do with other people.

While users exist in certain locations and times, they are able to request some tasks using speech recognition based on available resources and at the same time are free to carry out other functions. In this section, the users or objects location information will be communicated by the speech recognition approach.

In the Intelligent Environment research area, gesture and speech are becoming a new form of human interaction with the computer environment. Adding speech recognition and speech synthesis to the architecture of this Ubiquitous Computing environment can be crucial in improving the distribution of context information. Speech interaction also allows hands-free access to the Intelligent Environments, even when the user is away from the computer (Nieuwoudt and Botha 2002; Rebman, Aiken et al. 2002).

A prototype context agent was developed using Automatic Speech Recognition (ASR), which is called Speech Context Agent (SpeechCA), especially for the user to communicate the object's location to the environment. The cross platform Java Speech API (JSAPI) is used to implement speech synthesis and voice recognition. JSAPI supports command and control recognisers, dictation systems and speech synthesizers as shown in Figure 4.6.

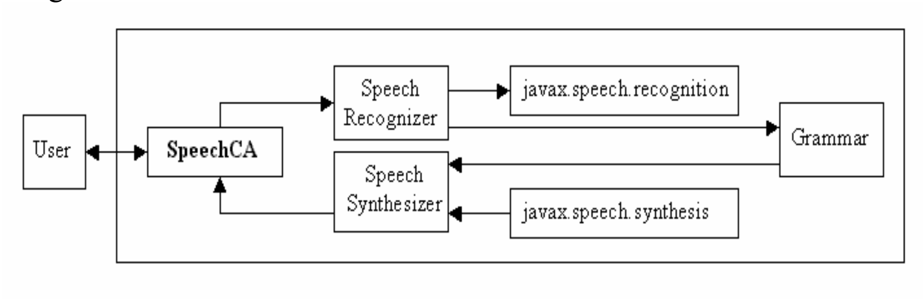


Figure 4.6 Block Diagram Speech Context-aware Prototype

SpeechCA needs a grammar to control the recognition and to inform the speaker what vocabulary and pattern will be recognised. A grammar contains a list of objects and instructions to generate query. The grammar has the advantage of making the recognition faster and more accurate.

4.5.1 The Use of Predicted User Location in SpeechCA Commands

A user can give a speech command to SpeechCA and SpeechCA can interpret the instruction using its "dictate" capability. The instruction will be interpreted as an SQL query to a location awareness history database. The result will be in text data type and it can be sent to a speech synthesizer as the Intelligent Environment reaction from this speech command.

The example below is the scenario of locating people using a speech context agent:

At 04.10 pm on 27 August 2004, John tried to locate Mary. He asked the SpeechCA as below:

John	**:**	Locate Mary, please …
SpeechCA	**:**	Mary was located at room 235 in DCS building, 5 minutes ago. The most probable location of Mary is in Room 235 in DCS or Room 103 in DE.

When a user gives the instruction *locate /name/*, SpeechCA will respond by generating a query concerning Mary's location to the location history and room database. It uses SQL to query his whereabouts, at a given time, and extends the queries to try to find an allocation logged with the minimum time difference in the set for the same day.

If the minimum time difference is close to zero, it means that SpeechCA found Mary's location and will respond with */name/ is located at room /r.Room/ in /r.Building/ building.* If it is not close to zero, SpeechCA's response will be */name/ is located at room /r.Room/ in /r.Building/ building, /difftime/ minutes ago.* Assume that if the different time is close to zero, the user is still in the same place. In an office environment, close to zero can be maximum 10 seconds for example:

```
# 5 minutes ago ….
Select r.Room, r.Building,
        difftime as min(time() -  h.time) ;
From RoomDB r, HistoryDB h  ;
Where     h.userid = 'teddy' and
          h.date = date() and  r.LocationID = h.LocationID ;
Order  by difftime
```

When SpeechCA finds that the minimum time difference is not close to zero, SpeechCA will try to find where the most probable location is in the log history database based on certain policy. For example, the policies for location checkpoints are:

1. **The same day of the week**
   (assume regular work schedules, to find Mary's location based on the history data of her location in almost the same time and same day of the week).
2. **All the days in a one week range**
   (to find Mary's location based on the history data of her location in almost the same time during a week).

It can be implemented by requesting the following query:

```
Select r.RoomNo, r.Building ;
From RoomDB r, HistoryDB h ;
Where r.userid = 'teddy' and ;
   r.LocationID = h.LocationID and ;
   (r.LocationID in {  ;
       Select h1.LocationID ;
       From HistoryDB h1 ;
```

```
        Where dayofweek(h1.Date) = day(Date());
            and (h1.Time <= Time() + 15 and ;
            h1.Time >= Time() – 15) } or ;
            h1.LocationID in {  ;
                Select LocationID  ;
                From HistoryDB h2  ;
                Where (daynumber (Date())- ;
                    daynumber(h2.Date) <= 8) and ;
                    (h2.Time <= Time() + 15 and ;
                    h2.Time >= Time() – 15) })
    Order By h.Date, h.Time
```

If the result delivers many hits, taking the most common location or the most recent can be considered. Based on the results of the query above, Intelligent Environment will then respond with: *The most probable location of /name/ is in Room / r.Room / in /r.Building/ {or Room in / r.Room / in /r.Building/}.*

By simple extension of the SQL query, further conditions can be discovered, e.g.:

- Find simple user information without aggregation from the location-history data.

  User: What is Mary's extension number?
  SpeechCA: Mary's extension number is 58179

  User: What is Mary's phone number?
  SpeechCA: Mary's phone number is 61258179

  User: What is Mary's occupation?
  SpeechCA: Mary's occupation is Computer System Administrator

- Find people if they have a meeting (two or more people in the same location, at the same time).

  User: Where did Mary and John meet last time?
  SpeechCA: They are in Room 235 at DCS building 5 minutes ago.

4.5.2 Finding the Nearest Object Using SpeechCA.

The Intelligent Environment in our implementation was designed to have similar behaviour to humans in response to user request. When a user gives the instruction *where is /deviceDB.name/'s printer?* the Intelligent Environment responds by giving the query where the printer location is to the SpeechCA. It uses SQL to query the where-about of the printer. For example:

User: Where is mclw's printer ?
(Local pronunciation of "mclw" is to say the letters: "emm-cee-ell-double you")[2]
SpeechCA: mclw's printer is located at Room 211 in DCS building

[2] The training of the speech recogniser includes such local vocabulary.

The query is:

***Select** r.RoomNo, r.Building*
***From** deviceDB d, roomDB r*
Where** d.Name = 'mclw' **and** d.type = 'Printer' **and
d.LocationID I = r.LocationID

***Entity:** DeviceDB*

DeviceID	LocationID	Type	Name	…
P1	121	Printer	mclw	…
P2	123	Printer	dcsHPps	…
P3	124	Printer	dcsKYO	…
…	…	…	…	…

Based on the results of the SQL query above, an Intelligent Environment will then respond by sending the following text: */deviceDB.name/'s printers located at room / roomDB.Room / in /roomDB.Building/ building* to the speech synthesizer as a part of SpeechCA.

The same responses will happen when a user requests a set of printers' locations in a certain location. For example:

User: Where are all the printers on level 2 of DCS building?
SpeechCA: All printers on level 2 of DCS are
dcshpps's is located at room 223 in DCS building
and dcskyo's is located at room 237 in DCS building
and mclw's is located at room 211 in DCS building

When the user gives the instruction *where are {all} printers on /varlocation/ in /varBuilding/ building?* the Intelligent Environment will respond by giving SQL query where all printers are in a given variable to the device and room database. It uses SQL to query the where-abouts of all the printers such as below:

Select d.Name, d.Level, r.RoomNO, r.Building ;
From deviceDB d, roomDB r ;
Where d.Type = 'Printer' **and** r.Building = 'DCS' **and**
r.LocationID = d.LocationID **and** r.level = 2 ;
Order by d.Name

The Intelligent Environment will then respond based on the set of SQL query results above, by sending the following text: */deviceDB.name/'s printers located at room /roomDB.Room / in /roomDB.Building/ building {and at room /roomDB.Room / in /roomDB.Building/ building}* to SpeechCA.

By moving around the Intelligent Environment, an authorised user could rapidly change their access to the relevant information, the availability of the communications and the computational resources.

To find the closest object (printer) to user location, the first thing to know is the absolute location of the user and the distance from the user to the object. However, this is not enough. When a coordinate location is available in an indoor environment, the wall/room, level of building, the queue job in the spool printer and the accessibility of the user to the printer may need to be considered. The term “closest” has many possible connotations. It can be a distance or may be the earliest time for the user to finish his task/request. However, it is possible that the closest printer is just on the other side of the wall but the user doesn't have access to it.

Thus, in the process of finding the closest printer, the variables that are required to be considered, could be:

a. The distance to the printer
b. The level of the building. The weight level can be used for distance calculation of a different level, such as

$$Weight_{level} = \begin{cases} = 0 & if \quad diff = 0 \\ = 5 & if \quad diff = 1 \\ = 10 & if \quad diff >= 2 \end{cases}$$

c. The dividing wall/room. The weight wall can be used for distance calculation to the different wall/room, such as:

$$Weight_{room} = \begin{cases} = 0 & if \quad diff = 0 \\ = 3 & if \quad diff = 1 \\ = 9 & if \quad diff >= 2 \end{cases}$$

d. The spooling of the printer. Time duration for the job sitting in the queue spool printer before it sends to the printer.
e. The degree of accessibility of the users.

The resources have two statuses: accessible and inaccessible. A user can access the resources, it may be because the resources are either not shared or shared but because the user doesn't have authorised access to it. Accessible resources should have shared status and the user authority to use it.

Assume that the user’s coordinate location is known as (18, 5, 15), by considering distance, walls/rooms, level of building, the queue job in the spool printer and the accessibility of the user to the printer to respond to the query below:

User: Where is the closest printer to me?
SpeechCA: the closest printer to you is /*d.Name*/ at /*r.roomNo*/ in / *r.Building*/ building.

When user gives instruction w*here is the closest /DeviceDB.type/ to me?* SpeechCA will first query all location printers in the area, by checking to the location database.

Entity: **Location1DB**

DeviceID	**X**	**Y**	**Z**
P1	22	15	12
P2	12	15	12
P3	12	0	25
P4	15	30	13

Based on the printer locations available, SpeechCA calculates the relative location by using the different weight level and wall/room level, such as below:

Entity: **Location 2DB**

DeviceID	X	Y	Z	Wlevel	Wrx	Wry
P1	22	15	12	0	0	0
P2	12	15	12	0	3	0
P3	12	0	25	5	3	3
P4	15	30	13	5	0	0

Then, that process is followed by the consideration of the accessibility of the printer and the duration of the print jobs on the spooler.

Entity: **Location3DB**

Device ID	X	Y	Z	Access	Qspool
P1	22	15	12	0	N/A
P2	12	18	12	1	5
P3	17	3	28	1	0
P4	20	30	13	1	5

Where Access = 0 (inaccessible - not share or share but has no authority)
Access = 1 (accessible - share and has authority to print)

Based on all considerations above, the closest printer can be found by creating the SQL query below:

```
Select r.RoomNo, r.Building, d.Name ;
From DeviceDB d ; RoomDB r ;
Where d.Type = 'printer' and r.LocationID = d.LocationID and
        d.DeviceID in {
            Select lw.DeviceID,
                    min((sqrt((lw.x-varx)^2 +(lw.y-vary)^2+(lw.z-varz)^2) ))
            From Location3DB lw ;
            Where lw.access = 1 } ;
order by r.RoomNo, r.Building
```

Based on the results of this query, SpeechCA responds by sending the following text: */DeviceDB.Name/ at /RoomDB.roomNo/ in /RoomDB.Building/ building* to speech synthesizer as the final result to the user query.

4.6 Location Scalability

Location Scalability discusses the structure of location information, object location of the user, devices and sensors, and how the location-aware application responds to the query that implied the user location. In responding to the user location query, an Active Office considers four variables, i.e.:

- *"location hierarchy"* which is the location structure of the environment,
- *"input query"* level which is the location level of a user who does the query,
- *"request query"* level which is the location level of the query in respect of the object being searched
- *"scope query"* which is the scope of "input query" and/or "request query".

In responding to the user location query, by default, the location information will be in the lower level of location of the user except where it is mentioned otherwise in the query. For example, John's location is in the north corridor of building A and he is asking the Active Office: *Where is Geoff?* The answer to this location query could be varied. However, in an Active Office, it will be translated into a single interpretation, which, by default should go to the same building and the same hierarchical/level of John's location, which is room/corridor level. The location hierarchy in this case for example is university level, the highest level, followed by building level and room/corridor level. Room and corridor has the same level, which is the lower level.

The question above in an Active Office can be interpreted as the present location of John is in the north corridor of building A and looking for the room/corridor location of Geoff in building A. The "input query", which is John's location, is in corridor level and the "request query" is in corridor or room level and the "scope query" is in building A, which is the default building of the current user location. When Geoff's location is found, the answer will be: *Geoff is in room E119*, if not then the answer will be: *Sorry John, Geoff is not in building A*. Since the query is in room level, the answer by default will be in room level as well.

When a user needs to expand the answer of the query to the building, university, city, country or continent level, the response will depend on the level of the query. The level of location should be indicated in the query. In the example below three questions each have ambiguous interpretation, whereas an Active Office needs a single interpretation to answer the query:

1. *I am in building A, where is Geoff?*
2. *In which building is Geoff?*
3. *I am in building A, in which room is Geoff?*

John: *I am in building A, where is Geoff?* This is a sample query where "input query" is in building level, the "request query" is, by default, following "input query", in building level and the "scope query" is not only in building A but also in any building of the university. The Active Office interprets that John is in building A, and wants to know in which building is Geoff? So, the Answer: *Geoff is in building B.*

However, in the question below, the Active Office has "input query", which is not defined in the query, by default, followed from a lower level of user location up to "request query" level, which is in room and building level, the "request query" is in building level only and the "scope query" is in any building of the university.

John: *In which building is Geoff?*
The Active Office interprets that John is in the north corridor of building A, and wants to know in which building is Geoff? The interpretation is different but the result is the same as above.

John: *I am in building A, in which room is Geoff?* This is a sample query where "input query" is in building level, the "request query" is, as indicated in the question, in room level and the "scope query" is in any building of the university. The Active Office interprets that John is in building A, and wants to know in which room and building is Geoff? Answer: *Geoff is in room xyz of the building B*

First, it will search in the current building and then, search the surrounding buildings. If Geoff is found in a different building, then the building reference has to be mentioned.

In an Active Office, location scalability can be shrinking and may not be considered when the location hierarchy is defined only in a single level or it is also possible to be extended to several levels as discussed on evaluation of location scalability in Section 9.10.3. By changing the query and location hierarchy, the response from the Active Office can be extended, for example the changes in the three queries above, from building level to university level, can expand the scope of the query and as a result, will expand the response in answering the query.

4.7 Discussion

The known and ignored issues are discussed in this section. The issues in relation to the use of location context-awareness are shown below:

a. The precision and amount of identification and location information can be obtained and made available by:
   - Desktop computer login and activity can be captured for user identity and location information.
   - The office environment is an indoor environment, there is no off the shelf GPS location, but wireless and wired LAN location are sufficient.
   - The use of very few special purpose sensors, in low-cost and adaptable setup experimental environment.
b. The results from the speech context agent are used to achieve what users would assess as effective and useful location-aware computing for their purposes.
c. The design parameter for the distributed software architecture and system architecture location: cheap and indoor means low precision information using existing information sources.
d. Unobtrusive for users with minimised privacy concerns: use own computing and information appliances (desktop, PDAs) little additional wiring, no badges, few cameras.
e. Common database tools are used for fusion of current sensor data and historical sensor data.
f. Compression/inference of history data (Deductive or extract database techniques can be used to reduce the growth of history data). The purpose of this technique is to make sure that the data will not grow without control.

This part ignored several issues such as:

a. Resource discovery, resource description, and zero configuration deployment of new resources such as printers.
b. Scalability of the database to thousands of sensors.
c. Extending the user location application to campus and further geographical scope.
d. Security: privacy and sensitivity of location information.

4.8 Summary

This chapter showed location awareness as a crucial part of the context-awareness mechanism for Ubiquitous Computing. How the Intelligent Environment determines user location is a complex problem. A user location model is described, with the model using various sensors to find, to detect and to predict user location in an Intelligent Environment.

The users' locations in an Active Office can be placed in three categories, i.e. precise location, proximate location, predicted location. The categories are based on sensor capability to sense the area and the use of history data. The priority order in deciding current user location is first precise location followed by proximate location, and then predicted location.

Experiments using WiFi and Bluetooth in determining proximate location in an Active Office have shown promising results in sensing user location. At present, WiFi seems to be the better way to determine precise user location by using proximate sensors rather than Bluetooth or IrDA. Results can be improved by developing interoperability between sensors to yield aggregated sensor data.

This chapter investigates user location; this approach can be used to determine equipment and user location in an Active Office.

Chapter 5

INSTANCE-BASED LEARNING METHODS FOR ESTIMATION OF SYMBOLIC USER LOCATION

This chapter introduces a novel algorithm for the location-awareness problem of estimating symbolic user location for indoor spaces using IEEE 802.11 (WiFi) wireless signals. The characteristic of the problem is that the signals fluctuate greatly, not only across perturbations in space, but also in time (diurnally), which leads to poor location estimation.

The ηk-Nearest Neighbour (ηk-NN) Algorithm is an instance-based learning algorithm which normalises the sample data set of the WiFi signal strength and signal quality to achieve the best correct result for symbolic user location at a room scale. Data normalisation is found to play an important role in determining the quality of the training data-set which has direct impact on the estimation result. The algorithm has been compared to other *k*-Nearest Neighbour (*k*-NN) and shows promising results.

5.1 Introduction

The variety of mobile devices such as Notebook, PDA, and Smart Phone in the market are still lacking in satisfactory location technology. Location-Aware computing, which promises accuracy, economic and easy deployment, always seems to be under construction. Numerous location models have been proposed in different domains and can be categorised into two classes, i.e. a symbolic (descriptive, hierarchical, topological) locations such as a city or a named room, and a coordinate (Cartesian, metric or geometric) location such as GPS, as described in Section 4.2.

As user location is an important part in Ubiquitous Computing, symbolic location is preferred to coordinate location to make daily communication easier on the more natural human location scale. However, coordinate locations can be converted into symbolic location.

The latest mobile devices (Notebook, PDA or Phone PDA) provide embedded wireless technology such as Bluetooth, IEEE 802.11 (WiFi) and mobile phone network (GSM/GPRS) to access location and network in the Ubiquitous Computing environment. In estimating symbolic user location, WiFi signals can provide less than a meter location precision. This could locate a user at the room scale in indoor spaces (home or office).

There are three main strengths in this work. First, real-life experiments and readings are provided to illustrate the fact that WiFi signals from Access Points can vary unpredictably, not only across perturbations in space, but also in time (diurnally). The reading of signal strength between Morning and Afternoon was found to be significantly different. It reached a difference of -33 dBm in a range of -99 to 0 dBm. This variation can clearly introduce errors in techniques based on un-normalised signal strengths. Second, this work proposes a simple but effective technique of normalising the WiFi signals readings and using a *k*-Nearest Neighbour strategy to improve the localisation accuracy. Third, as the quality of the training data-set plays an important role in estimating symbolic user location, this work proposes the use of the Boolean MaxMin algorithm to evaluate the quality of the training data-set and proposes the normalisation of the signal strength and signal quality of the training data-set. It would give a determination to produce a correct location as a result.

The machine learning approach, in particular instance-based learning methods, is used to estimate user location. It has two processes: *learning and estimating*. The problem in the learning process is how to define a good quality training data-set to get a good result in estimating user location? This follows the question in the estimating process: What kind of approach is needed to achieve the most correct results?

To answer those questions, two algorithms were proposed. The ηk-Nearest Neighbour algorithm is proposed to be used in instance-based learning methods of the machine learning algorithm. This approach used normalised training data, which is a transformation applied uniformly to each element in a set of WiFi's signals so that the training data-set has some specific property. The property is signal determination in room scale in this case. First the Boolean MaxMin algorithm was used to resolve the problem of how to define good quality data before the learning process. Second, to achieve a good result, the maximum number of locations that appeared from the first nearest ten of the the ηk-Nearest Neighbour Algorithm is proposed for use in the estimation process. From our experiments, the recommended numbers to get the highest accuracy should be greater than four. However the bigger the number is not a guarantee to getting the best correctness, it depends on the sensitivity of the area being examined. Ten is one of the recommended numbers that reach the best correctness in this case.

This techniques focuses on the problem of discrete localisation which resolves the user's location to a specific room, rather than the continuous localisation goals of previous work such as RADAR, Nibble, Ekahau, CMU-TMI (Bahl and Padmanabhan 2000; Castro, Chiu et al. 2001; Smalagic and Kogan 2002; Ekahau 2005). This technique is not based on triangulation, interpolation or mapping algorithm such as in (Smalagic and Kogan 2002; Hightower and Borriello 2004), but it uses a 'trapping method' to trap radio signals that lead to symbolic user location.

In this experiment, the machine learning algorithm focus on the 6 closest WiFis' access points among 21 access points covering the three connected building. The access points come from different brands and also have different standards (802.11 b, b+ and g). The hot-spot areas from the access points overlap and the two attributes of signal strength and signal quality were measured at many points in a building, representing multivariate data. Generally, the signals are available from WiFi driver software. The user used a Dell Inspiron 600m notebook with Dell TrueMobile^{TM} Wireless LAN adapter and a Linux Fedora Core 1 operating system. The program to measure signal strength and the machine learning algorithm is written in Python language.

In general the process in the estimation of user location can be described as follows:

- collect the WiFi's signal strength and quality data
- develop algorithm to estimate user location
- test the algorithm and evaluate the result

In the Active Office, machine learning is implemented using the *k*-Nearest Neighbour algorithm (Brumit, Meyer et al. 2000) which is improved by normalising the data set by using the standard deviation. The result of this approach is quite promising, the more the algorithm learns the better the result it gives.

This chapter presents an overview of instance-based learning and *k*-Nearest Neighbour in Section 5.2, followed by proposed algorithms, the *ηk*-Nearest Neighbour algorithm, the Boolean MaxMin algorithm to evaluate the quality of the training data set and finding the best *k* (maximum common value) to achieve the best correct result in the estimation of symbolic user location. Discussion and analysis of the result of the algorithm is also presented, followed by a summary and further study which may arise from this work.

5.2 Machine Learning for Location Awareness

The Intelligent Environment requires systems that are able to learn, adapt and interact intelligently with the environment in which users operate in ‘situated intelligence’. The behaviour of these systems could be characterised by intelligent algorithms, especially the Intelligent Environment’s tasks that involve some kind of learning and adaptation.

Machine learning which deals with programs that learn from experience is implemented to estimate user location in the Intelligent Environment, which improves or adapts the performance of a certain task or group of tasks over time. The goal of machine learning in this implementation is the design of agents/programs that learn and/or discover an algorithm that is able to estimate user location, also automatically improving their performance on certain tasks and/or adapting to changing circumstances over time (Mitchell 1997; Leeuwen 2002).

5.3 Training: The Description of the Learning Process

An on-line supervised learning scenario is used in which the program is fed examples and must predict the label of every next example before a ‘teacher’ gives the answer. In on-line learning, the program is given examples one by one, and it recalculates its *hypothesis* of what it has learnt after each example. Examples are the signal quality and signal strength that are randomly picked up from the access point in the whole building (a random source), according to a normal probability distribution. An on-line scenario is interactive learning, in which case new examples are supplied depending on the performance of the program on previous examples (Leeuwen 2002). An off-line learning scenario can also be used, in which the program receives all examples at once and not in an interactive mode.

Training scenarios are typically finite but, in this work, are designed to continuously learn to improve and adapt forever. In other words, it is an *inductive inference,* a program fed an unbounded amount of data.

When the data was tested, the reinforcement learning method was also implemented. The inputs come from unpredictable WiFi signal strength and signal quality, when positive or negative feedback is given at the end of every small sequence of learning steps, and which are part of the process of learning an optimal strategy. If in our approach it were wrong three times successively, reaching -3 of the feedback, the data that causes the problem will be removed from the learning data-set and the hypothesis modified.

At the learning stage, signal strength and signal quality gather and put together all possible information on numerous variables to make location estimation. On an off-line training method a multivariate data-set is used, in which the set of observed values of several variables is taken from the same subjects or experimental units. Experimental units are referred to as *occurrence* or *records* and variables as *fields*.

When the online training method is being used, the algorithm learns the unknown *concept*, i.e. a minimum and maximum of the Boolean function, which classified signal examples. The program first estimates the user location in the building based on its current *hypothesis*. Following that, it compares the hypothesis and the concept, if the hypothesis is right then no changes are made, but if it is wrong, the program subsequently revises its hypothesis based on the knowledge of the example so far. The process continues until the hypothesis is consistent with the concept by suitable choice of learning. Any correction of a hypothesis is the *classifier* of the concept.

5.4 Instance-Based Learning and the *k*-Nearest Neighbour

Instance-based learning methods are conceptually straightforward approaches to approximate real-valued or discrete-valued target functions. Learning these algorithms consists of simply storing the presented training data set. When a new query instance is encountered, a set of similarly related instances is retrieved and used to classify or estimate the new query instance.

Instance-based learning can be designed to perform approximation well by constructing only local approximation to the target function that applies in the neighbourhood of the new query instance without processing the entire instance space. This is the significance of the advantages of instance-based learning, especially when the target function is very complex, it can still be described by the collection of less complex local approximations. The advantages of instance-based methods include the ability to model a complex target function from a collection of less complex local approximations and the fact that the information present in the training example is never lost because the examples themselves store explicitly. The main practical difficulties include the efficiency of labelling new instances. All processing is done at the query time rather than in advance, difficulties exist in determining an appropriate distance metric for retrieving related instances, especially when examples are represented by complex symbolic descriptions and the negative impact of irrelevant features on the distance metric (Mitchell 1997).

The disadvantages of this approach are

a. The computation of classifying new instances can be very high, due to the fact that nearly all computation takes place at classification time rather than when training examples are first encountered.
b. When attempting to retrieve similar training examples, this approach considers all attributes of instance space. If the target concept depends on only a few of the many available attributes, then instances that are truly most 'similar' may well be a large distance apart.

The *k*-Nearest Neighbour family is an instance-based algorithm for approximating real-valued or discrete-valued target functions, assuming instances correspond to points in an n-dimensional Euclidian space. The target function value for a new query is estimated from known values of the *k*-Nearest training examples.

Numerous algorithms for *k*-Nearest Neighbour queries are proposed. This type of query is extensively used in geographical information systems, shape similarity in image databases or pattern recognition. A majority of the algorithms are aimed at m-dimensional objects and are based on utilising one of the variations of multidimensional *vector or metric index structures* (Kolahdouzan and Shahabi 2004).

In the *vector index structure* approach, Roussopoulos et al. proposed a branch-and bound R-tree traversal algorithm to find the nearest neighbour of a query point (Roussopoulos, Kelly et al. 1995). The main disadvantage of this approach is the depth-first traversal of the index that incurs unnecessary disk access. Korn et al. proposed a multi-step *k*-Nearest Neighbour search algorithm (Korn, Sidiropoulos et al. 1996) and Seidl and Kriegel proposed an optimal version of this multi-step algorithm by incrementally ranking queries on index structure (Seidl and Kriegel 1998). The disadvantage of this approach is that the number of candidates obtained in the filter step is usually much more than necessary, making the refinement step very expensive.

Metric-index structure approaches are also based on a filter and refinement process, but as opposed to vector index structures, the indexing and filtering of the objects considers their metric distance. Berchtol, Ertl et al. proposed pre-calculating, approximating and indexing the solution space for the nearest neighbour problem in m-dimensional spaces (Berchtold, Ertl et al. 1998). This approach is only appropriate for the first nearest neighbour problem in high-dimensional spaces. Jensen, Kolarvr et al. proposed a data model and definition of abstract functionality for nearest neighbour in spatial network database (Jensen, Kolarvr et al. 2003). They use algorithms similar to Dijkstra to calculate the shortest distance from query to an object on-line. Papadias, Zhang et al. proposed a solution for nearest neighbour queries in a network database by introducing an architecture that integrates network and Euclidian information and captures pragmatic constraints (Papadias, Zhang et al. 2003), and Kolahdouzan and Shahabi proposed pre-calculating the network Voronoi polygon and pre-computing network distance (Kolahdouzan and Shahabi 2004). The advantages of this approach are that it offers a method that finds the exact distance in networks while the architecture can support other spatial queries like range search and closest pairs.

5.5 The ηk-Nearest Neighbour Algorithm

The *k*-Nearest Neighbour algorithm is used in the instance-based learning method to classify and then estimate symbolic user location in the Ubiquitous Environment. This algorithm assumes all instances correspond to points in the *n*-dimensional space $\Re^n$ of WiFi's signal strength and signal quality.

The proximate sensors such as Bluetooth and WiFi scanner sensors are not accurate in estimating the user location. In the case of the WiFi sensor, the sensor has signal strength, signal quality and noise. Noise is relatively a constant variable in the whole building, this is not significant for the estimation of user location. The most significant variable in the estimation of symbolic user location is signal strength, but the problem is that it tends to fluctuate greatly. This makes it difficult to correctly estimate symbolic user location.

This problem occurs when these measures change even though there are no objects moving in the hot-spot area. Figure 5.1 shows how the signal changes. The dots represent signal strength when it is measured in point b, while the signal itself has the possibility of interfering up to points a and c.

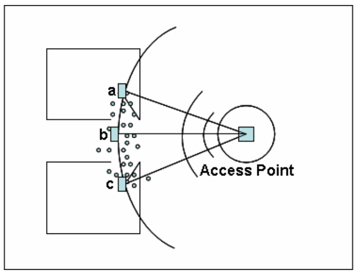


Figure 5.1 The Changing of the Signal Strength

The nature of radio communication at microwave frequencies requires line of sight for optimal performance. This means that between two or more antennas there should be an unobstructed view which is impossible in an indoor environment. The possible causes of the changing WiFi's signal strength, signal quality and noise (Flickenger 2002) are as below:

- Objects that absorb microwave signals, such as trees, earth, brick, plaster walls, and people.
- Objects that reflect or diffuse signals, such as metal, fences, mylar, pipes, screens, and bodies of water.
- Sources of 2.4 Ghz noise, such as microwave ovens, cordless phones, wireless X-10 automation equipment, and other 802.11 networks.

The estimation of user location is not always correct and, to minimise this problem, the machine learning technique could be chosen. In estimating the user location, the approach is to develop an instance-based learning algorithm that is used with the *k*-Nearest Neighbour algorithm to deal with multivariate data of WiFi's signal strength and signal quality. The nearest neighbours of a WiFi's instance can be defined using the standard Euclidian distance.

More clearly, let an arbitrary instance x be described by the feature vector:

$$f(x) = (a_1(x), a_2(x), \ldots a_n(x),\ a_{n+1}(x))$$

where $a_r(x)$ denotes the value of *r*th WiFi's signal and $a_{n+1}(x)$ is the symbolic location (room-no) attribute of instance *x*. In the case of 6 access points covering the building, *r*th between 1 and 6 is a signal strength attribute and *r*th between 7 and 12 is a signal quality attribute of instance *x*, therefore *n* is 12. The example of signal strength and signal quality from 6 access points can be seen in Table 4.2. Then the distance between two instances x_i and x_j is defined to be $\delta(x_i,x_j)$, where

$$\delta(x_i, x_j) \equiv \sqrt{\sum_{r=1}^{n} (a_r(x_i) - a_r(x_j))^2}$$

In nearest-neighbour learning the target function may be either discrete-valued of off-line or on-line data. Consider learning the discrete-valued target function of the form $f : \Re^n \to V$, where *V* is the finite set (v_1, v_2 …,v_s). The *k*-Nearest Neighbour algorithm for approximating a discrete-valued target function is shown in Table 5.1, the value $f(x_q)$ returned by this algorithm as its estimate of $f(x_q)$ is the most common user location value of *f* among the *k* training examples nearest to x_q. If *k*=1 is chosen, then the 1-Nearest

Neighbour algorithm assigns to $\bar{f}(x_q)$ the value $f(x_i)$ where x_i is the training instance nearest to x_q. For larger values of *k*, the algorithm assigns the most common value among the *k*-Nearest training examples.

The estimation of symbolic user location by calculating the minimum of the *k*-Nearest Neighbour is not precise, it can be improved by adding the functionality of the algorithm to find the list of the first ten (*k*=10) of the locations closest to the target where the symbolic user location is the most common that appears on the list. Table 5.1 shows the complete step of the algorithm.

Table 5.1 The *k*-Nearest Neighbour Algorithm for Estimating a User Location Valued Function $f : \Re^n \to V$ Using WiFi's Signal Strength and Signal Quality

Training algorithm:

- Let $a_1(x), a_2(x) \ldots a_{13}(x)$ denote the list of examples data-set of instance *x* that contains signal strength ($a_{1\text{-}6}$), signal quality ($a_{7\text{-}12}$) and location (room-no) (a_{13}) of instance *x*. Let $a_1(x)$ to $a_6(x)$ represent signal strength from access point 1 to access point 6, and $a_7(x)$ to $a_{12}(x)$ represent signal quality from access point 1 to access point 6. Let $a_{13}(x)$ be a room name where instance *x* is measured. For each training sample $(x, f(x))$ that contains signal strength, signal quality and location (room-no), add the sample to the list training data-set.

Estimate algorithm:

- Given a query instance x_q to find the estimated location, let $a_1(x_q), a_2(x_q) \ldots a_{12}(x_q)$ denote the list of data query instance x_q value that contains signal strength ($a_{1\text{-}6}$), signal quality ($a_{7\text{-}12}$) and let the estimation location be denoted in $a_{13}(x_q)$.
- Finding the first ten (*k*=10) nearest Euclidian distances signals of the *k*-Nearest Neighbour:

  $$\bar{f}(x_q) \leftarrow \underset{v \in V}{\arg\min}\, \delta(x_q, x_v), \quad \forall x_v \in V$$

  where $\delta(x_q, x_v) = 0$ if $x_q = x_v$ and $\delta(x_q, x_v) > 0$ otherwise.
- The user location is given by the highest number (most common value) of locations that appear from the nearest ten (*k*=10) Euclidian distances of the *k*-Nearest Neighbour.
- Return the user location as a $a_{13}(x_q)$

The WiFi's signals' data fluctuates greatly, which makes it difficult to find the determination between rooms. A problem occurs when user location is estimated and the signal can lead to more than one room. The fluctuating signal greatly increases the probability of getting the wrong estimation. Finding the determination eliminates this problem. From the experiments, the normalisation of the signal strength and signal quality data produces significant determination between rooms, closely approaching the correct symbolic estimation location. The data normalisation for this machine algorithm uses mean and standard deviations (Brumit, Meyer et al. 2000) in room scale. These approaches can significantly reduce noise but it does not mean there is no noise at all because of the fluctuating quality of the WiFi's signals.

Table 5.2 ηk-Nearest Neighbour Algorithm: The Algorithm to Estimate a User Location Valued Function $f : \Re^n \to V$ Using Normalisation (η) of the WiFi's Signal Strength and Signal Quality

Training algorithm:

- Let $a_1(x), a_2(x) \ldots a_{13}(x)$ denote the data-set of measurement instances x that contains signal strength, signal quality and location (room-no) of instance x. Let $a_1(x)$ to $a_6(x)$ represent signal strength from access point 1 to access point 6, and $a_7(x)$ to $a_{12}(x)$ represent signal quality from access point 1 to access point 6. Let $a_{13}(x)$ be a room name where instance x is measured. For each training sample *(x,f(x))* that contains signal strength, signal quality and location (room-no), add the sample to the list training data-set.
- Group the examples data $a(x)$ by room-no $a_{13}(x)$ and calculate mean $\bar{a}$ of the signal-strength and signal quality for each group.

  $$\bar{a}_{(j)}^{(k)} = \frac{\sum_{x_i} a_j(x_i)}{n(k)},\ \forall x_i \text{ such that } a_{13}(x_i)=R_k,\ \forall j \text{ such that } j \in [1,12].$$

  where:
  - $n(k)$ denotes the number of instances of x (number of measurements) in the same room R_k.
  - a_j denotes the signal-strength or signal quality from WiFi access point 1 to 6.
  - R_k denotes location (room-no).
  - $a_j(x_i)$ denotes the value of jth signal strength or signal quality attribute of instance x_i.
  - $\bar{a}_{(j)}^{(k)}$ denotes the mean value of jth signal strength or signal quality over all instances in room R_k.
- Calculate the standard deviation σ of the signal-strength and signal quality for each group.

  $$\sigma_{(j)}^{(k)} = \sqrt{\frac{\sum_{x_i}\left(a_j(x_i) - \bar{a}_{(j)}^{(k)}\right)^2}{n(k)-1}},\ \forall x_i \text{ such that } a_{13}(x_i)=R_k,\ \forall j \text{ such that } j \in [1,12].$$

  where:
  - $\sigma_{(j)}^{(k)}$ denotes the standard deviation jth signal strength or signal quality over all instances averaged in room R_k.
- Normalisation of the training-data-set based on mean and standard deviation above.

  $$\eta_{(i,j)}^{(k)} = \frac{a_j(x_i) - \bar{a}_{(j)}^{(k)}}{\sigma_{(j)}^{(k)}},\ \text{such that } a_{13}(x_i)=R_k,\ \forall j \text{ such that } j \in [1,12].$$

  where:
  - $\eta_{(i,j)}^{(k)}$ denotes the normalisation of the value of jth signal strength or signal quality attribute of instance $a_j(x_{(i)}^{(k)})$.

- For each training example *(x,f(x))*, add the example to the list training data-set of the normalisation of the data-set (x).

Estimate algorithm:

- Given a query instance x_q to find the estimated location, let $a_1(x_q), a_2(x_q)$ ... $a_{12}(x_q)$ denote the list of data query instance x_q value that contains signal strength ($a_{1\text{-}6}$), signal quality ($a_{7\text{-}12}$) and let the estimation location be denoted in $a_{13}(x_q)$.
- Normalise the instance x_q which is $\eta_j^k(x_q)$ against each room R_k in turn using mean $\bar{a}_j^k$ and standard deviation $\sigma_{(j)}^{(k)}$.
- Select the measurement instances x_i for which the distance $\delta(\eta^k(x_q), \eta^k(x_i))$ is the 10 smallest distances.
- The estimated room number for the query x_q is the majority room number $a_{13}(x)$ for this set of 10 nearest measurement instances.

Note: in the event of tie for the majority count the lower-numbered room is taken as an arbitrary tie-breaker.

The use of Euclidian distance without data normalisation in finding user location is very straight forward, i.e. calculate the first ten nearest neighbours and find the maximum number of the location as the estimation location. However, the use of Euclidian distance with normalisation is quite a different approach. It needs to consider the group or the classifier for a new instance to normalise and find to which classification the new instance belongs. It is based on mean and standard deviation for each group. It is then followed by calculation of the first ten (k=10) nearest neighbours and finding the maximum number of locations as the estimation location.

The operation of the k-Nearest Neighbour algorithm for the case where the instances are points in a two-dimensional space and where the target function is a user location which can be deduced from:

a. the minimum of the k-Nearest Neighbour (k-NN)
b. the minimum of the ηk-Nearest Neighbour (ηk-NN)
c. the maximum number of locations that appear from the nearest ten Euclidian distances of the k-Nearest Neighbour (k-NN(10)) as shown at Table 5.1.
d. the maximum number of locations that appear from the nearest ten Euclidian distances of the ηk-Nearest Neighbour(ηk-NN(10)) as shown at Table 5.2.

The results were analysed and discussed in Section 7.

The ηk-Nearest Neighbour algorithm is easily adapted to approximate continuing value target functions. To accomplish this, the algorithm calculates the mean value of the ηk-nearest training examples rather than that of their most common value.

5.6 The Algorithm to Evaluate the Training Data Set

The performance of the k-Nearest Neighbour algorithm can be determined by two variables: the sample training data set and the algorithm itself. The algorithm has been studied in Section 5. This part will discuss how to determine a good quality training-data set. The training data-set is good quality input data if there is a determination that narrows it to a single location.

Table 5.3 The Boolean MaxMin Algorithm to Determine the Quality of the Training Data Set

The Boolean MaxMin algorithm:

1. For each training example *(x,f(x))* add to the list data-set evaluation
2. Grouping the data-set based on the classifier, which is the location (in room-no based).
3. Finding the maximum and minimum of the instance signals. Let $x_1, x_2, \ldots x_k$ denote the k instance of signal strength and signal quality, where $x_1 \ldots\ldots x_k \in C$, and C is a classifier for the user location in C space.

   $\bar{f}(x_i) \leftarrow \min(x_i)$ for $i \in k$ and $x_i \in C$ and

   $\bar{g}(x_i) \leftarrow \max(x_i)$ for $i \in k$ and $x_i \in C$
4. Finding the maximum of the minimum (maximin) signal from location classification (room based).

   $\bar{\bar{g}}(x_i) \leftarrow \max(\bar{f}(x_i))$ for $i \in k$ and $x_i \in C$
5. Finding the minimum of the maximum (maximin) signal from the location classification (room based).

   $\bar{\bar{f}}(x_i) \leftarrow \min(\bar{g}(x_i))$ for $i \in k$ and $x_i \in C$
6. The data set is a qualified data set if and only if a minimum one ‘false’ value is found in the target Boolean function. The target function is a Boolean value achieved by comparing the maximin and the minimax of the WiFi’s signals instance data-set for training.

$$\beta(x_i) = \begin{cases} \bigcup_{i=1}^{k}((\bar{\bar{f}}(x_i) - \bar{\bar{g}}(x_i) < 0 \Rightarrow' false' \\ \bigcup_{i=1}^{k}((\bar{\bar{f}}(x_i) - \bar{\bar{g}}(x_i) > 0 \Rightarrow' true' \\ \\ \quad otherwise \Rightarrow' false' \end{cases}$$

To date, to the best knowledge of the author no literature exists which discusses the method of evaluating the training data-set. In this work, we design an algorithm to evaluate the training data-set, a Boolean MaxMin algorithm, to determine the quality of the training data set (Table 5.3).

It can be seen from the algorithm above that once the ‘false’ value is found from the target function in the training data, then the data set will be a good quality training data set.

5.7 Discussion

In this part, the result of the four variations of *k*-Nearest Neighbour algorithms that are explained in Section 5.5 will be discussed, followed by a discussion of the Boolean MaxMin algorithm in the evaluation of the training data set.

5.7.1 The Result of the Four Variations of *k*-Nearest Neighbour Algorithms

The four variations of *k*-Nearest Neighbour algorithms use the training data-set that is collected by measuring the WiFi's signal strength and signal quality in room E213, room E214, Corridor 1, Corridor 2 and the long Corridor 3 in a building with 6 access points for the learning process. For the estimation process, the user location is estimated by measuring the user's current signal strength and signal quality and comparing it with the training data set. The algorithm returns the estimated user location, and calculates the percentage of correctness over time.

The algorithm *k*-NN and *ηk*-NN were tested together. The process started at 19:38:55 for 14 hours and produced 104988 measurements. The results were of 79.59% and 90.30% correctness respectively. The algorithm *k*-NN(10) and *ηk*-NN(10) were also tested together. The process started at 19:37:31 for 14 hours on a different day. It produced 69128 measurements. The results were of 80.89% and 95.82% correctness respectively. Details are shown in Table 5.4.

Table 5.4 The Comparative Results of the Four Algorithms for 14 Hours Measurements

No. of Measurement	Location					Algorithm
	E213	E214	Cor1	Cor2	Cor3	
104988	79.5867	2.4706	15.9098	1.3075	0.72440	*k*-NN
104988	90.2988	0.6464	4.84960	3.1270	1.07730	*ηk*-NN
69128	80.8949	1.07583	0.68901	0.0583	0.08737	*k*-NN(10)
69128	95.8165	0.0376	3.53110	0.6148	0	*ηk*-NN(10)

In general, as shown from Figures 5.2 to 5.5, machine-learning algorithms work very well. The longer the algorithm was tested the more stable the result. During the testing of *k*-NN and *ηk*-NN, other WiFis' signals that scan the rooms were found and the location category for this signal data was the 'Other' room. The numbers in this category are very small so it can be disregarded.

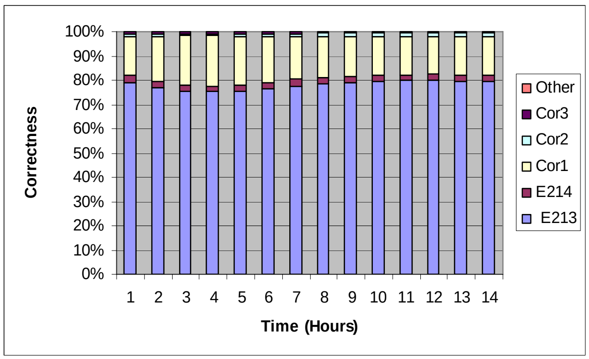


Figure 5.2 The Minimum of the *k*-Nearest Neighbour

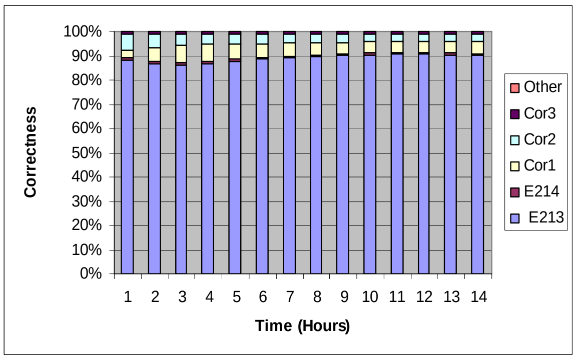


Figure 5.3 The Minimum of the *ηk*-Nearest Neighbour

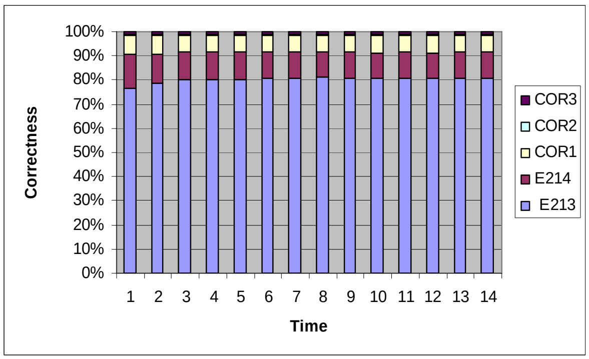


Figure 5.4 The Maximum Number of Locations from the Nearest Ten of the *k*-Nearest Neighbour

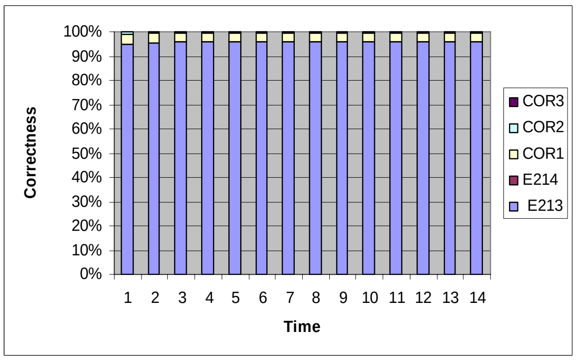


Figure 5.5 The Maximum Number of Locations from the Nearest Ten of the *ηk*-Nearest Neighbour

5.7.2 The Boolean MaxMin Algorithm

The purpose of the Boolean MaxMin algorithm is to determine the quality of the training data-set. If it is a good quality data set it can be then fed to the learning stage on the machine learning algorithm. The training data-sets are first collected by measuring each point (about a meter) six times within a room or a corridor. In this case, the signals were measured in the surrounds of the 2 rooms (E213 and E214) and 3 corridors (Cor1, Cor2 and Cor3), which have 66 points, hence the training data-set has 396 data measurements. Following this, the maximum and the minimum of the signal strength and signal quality were calculated based on room scale for two types of data sets: the regular data set and the normalisation data set. Table 5.5 shows the results of the calculation of the maximum of the signals without data normalisation and the last data table shows the minimum of the maximum (MiniMax) from access point 1 to access point 6.

The same process used to find the maximum of the signal strength and signal quality are also used to calculate the maximum of the minimum (MaxiMin) of the signal strength. Next, the MiniMax and the MaxiMin for every signal strength and signal quality were compared. The Boolean value was used for this purpose. If the signal strength of the MaxiMin from access point 1 was *less than* the signal strength of the MiniMax from access point 1 then the 'false' value is given, otherwise it is a 'true' value. The same process was also followed for all signal strengths and signal qualities from access point 1 to access point 6 as explained in the Boolean MinMax algorithm in Section 4.

If all the Boolean values of the MaxiMin and MiniMax are 'true' (shown in Table 5.6), it means that the training data-set is a poor quality training data set. It means that given an instance x between the MaxiMin and MiniMax data to the *k*-Nearest Neighbour algorithm, it would easily give a wrong location as a result.

Both algorithms, the Boolean MinMax and *ηk*-Nearest Neighbour, can be used by a mobile's client application to estimate the user location and as based of a data distance. This can be used to find relevant localisation information. Dar, Franklin et al., and also Ren and Dunham proposed a Manhattan distance and Further Away Replace (FAR) to be used as the data distance especially for cache replacement policies (Dar, Franklin et al. 1996; Ren and Dunham 2000). The data distance can affect the performance of local queries, depend on the mobile client's movement and query patterns.

Table 5.5 Example of the Maximum of the WiFi's Signal Strength and Signal

Signal Strength						Signal Quality						Room No
AP1	AP2	AP3	AP4	AP5	AP6	AP1	AP2	AP3	AP4	AP5	AP6	
-48	-79	-84	-99	-82	-73	1.391	0.348	0.696	0	2.087	2.435	Cor1
-45	-76	-93	-99	-60	-60	1.391	0.348	0.696	0	2.087	2.435	Cor2
-56	-80	-99	-99	-80	-71	1.391	0.348	0	0	2.087	2.435	E213
-52	-90	-83	-99	-81	-74	1.391	0.348	0.696	0	2.087	2.435	E214
-56	-90	-99	-99	-82	-74	1.391	0.348	0	0	2.087	2.435	MiniMax

Table 5.6 The Boolean MaxiMin and MiniMax for the Analysis of WiFi's Signal Strength and Signal Quality

Signal Strength						Signal Quality						Max/Min
AP1	AP2	AP3	AP4	AP5	AP6	AP1	AP2	A3	AP4	AP5	AP6	
-56	-90	-99	-99	-82	-74	1.391	0.348	0	0	2.087	2.435	Maximin
-70	-99	-99	-99	-99	-87	1.391	0	0	0	0	2.435	Minimax
True	True	True	True	True	True	True	True	True	True	True	True	True

Table 5.7 Example of the Minimum of the Normalised WiFi's Signal Strength and Signal Quality

Signal Strength						Signal Quality						Room No
AP1	AP2	AP3	AP4	AP5	AP6	AP1	AP2	AP3	AP4	AP5	AP6	
-3.713	-0.967	-0.494	0	-0.668	-2.770	-4.007	-1.087	-0.519	0	-0.687	-4.007	Cor1
-1.873	-2.040	-0.174	0	-1.545	-2.968	0.985	-5.570	-0.174	0	-1.898	-0.985	Cor2
-5.917	-0.613	0.000	0	-1.164	-0.865	-0.993	-0.733	0.000	0	-1.223	-0.964	E213
-2.585	-0.435	-0.222	0	-1.120	-2.330	0.986	-0.441	-0.239	0	-1.167	-0.986	E214
-1.873	-0.435	0.000	0	-0.668	-0.865	0.986	-0.441	0.000	0	-0.687	-0.964	MaxiMin

Table 5.8 The Boolean MaxiMin and MiniMax for Analysis of the Normalised WiFi's Signal Strength and Signal Quality

Signal Strength						Signal Quality						Max/Min
AP1	AP2	AP3	AP4	AP5	AP6	AP1	AP2	AP3	AP4	AP5	AP6	
0.862	1.484	0	0	1.323	1.670	-0.993	0.174	0.000	0	0.511	-0.986	MaxiMin
-1.873	-0.435	0	0	-0.668	-0.865	0.986	-0.441	0.000	0	-0.687	-0.964	MiniMax
True	True	True	True	True	True	False	True	True	True	True	False	False

5.7.3 Finding the Best *k* (Maximum Common Value) to Achieve the Maximum Correct Result in the Estimation of Symbolic User Location

Section 7.1 shows that the ηk-Nearest Neighbour is the best algorithm among the four of the *k*-Nearest Neighbour algorithms. However, to achieve the maximum correct result of the ηk-Nearest Neighbour strategy, it needs a proper *k* value for the most common value in estimation of user location. The following experiments were performed in finding the best *k* value for the ηk-Nearest Neighbour algorithm:

1. Measurement of training data at different times.
   This used three types of WiFi signal strengths and signal qualities in the morning (9am), afternoon (1pm) and late afternoon (7pm).
2. Measurement of signal for estimation of user location in a building.
   This estimated a user's location using WiFi signal strengths and signal qualities at six arbitrary points, for 14 hours testing each point, in certain locations in a building.
3. Evaluated the most common value of *k* where *k* = 1, 2, 3, ...
   This is tested for *k*=1 to 11 for 14 hours measurement each, from 6 access points and 6 points (P1 to P6 in Figure 5.6). The correctness to accept the result shall be greater than 90%. Because of the observed continuing fluctuation in correctness with increasing *k*, we made a pragmatic choice of *k* = 10, well above the region k = 4 where correctness dropped below 90%.

In the *nk*-NN algorithm, *k* = 10 indicates the most frequent room number that appears in the list of ten locations which is closest to the target location (most common value). In that case, it has several same values; the algorithm will sort on the alphabetical order of the room name and choose the first one. For example, *k* = 10, and in the list of the first ten nearest locations 3 in room A, 3 in room B, 3 in room C and 1 in room D, then the estimation will goes to the first room, which is room A.

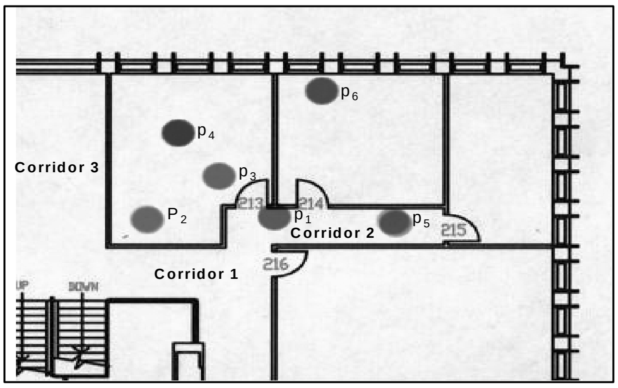


Figure 5.6 The Arbitrary Six Points at Which Measurements Were Taken in a Building

In this experiment, six points have been measured in a building as shown in Figure 5.6. The building has 7 access points, but the algorithm will only be using the first 6 points that provide the best signal strengths and signal qualities.

Figure 5.7 shows the fluctuation of the most common value of *k* = 1, 2, 3, … , 11, using the *nk*-NN algorithm. To find the most stable result, each process from *k*=1, 2, 3, …, 11 is tested for 14 hours on the estimation of user location using WiFi signal strength and signal quality. Points 1, 2, 3 were found to fluctuate in the results, whereas points 4, 5, 6 have stable results for nearly 100% of correctness.

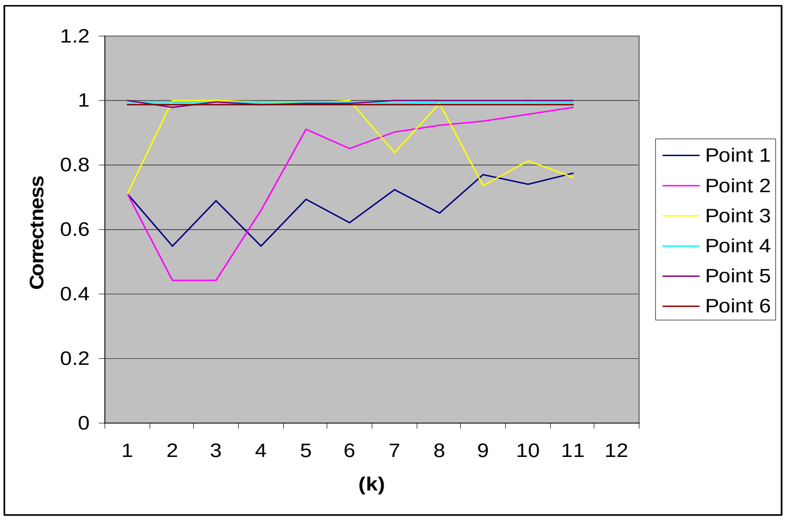


Figure 5.7 Fluctuation of the Most Common Value of *k* =1, 2, 3, … , 11, Where Each was Processed for 14 Hours on the Estimation of User Location Using WiFi Signal Strength and Signal Quality

From the experimental results, it can be seen that the results can be classified into two kinds of zones involved in estimating user location using WIFI signals, i.e.

- Noise zone, which is the zone with fluctuating results that make estimation difficult
- Stable zone, which is the zone with stable results that may be easily estimated

In Figure 5.6, points 1, 2, and 3 fluctuate between 0.4 and 1, which is in the noise zone and points 4, 5, and 6 fluctuate between 0.96 and 1, which is in the stable zone. This figure shows that the noise zone is usually located between rooms and/or in corridors near the edge of rooms or corridors. The stable zone is usually located in the central part of the room, which has good determination to other rooms/locations. These types of points have a stable result during the estimation process. The locations of the access points also have influence on the sensitivity of the location.

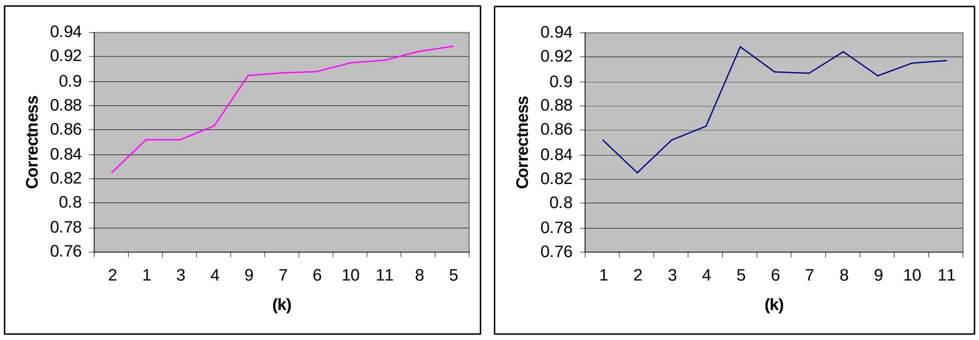


Figure 5.8 The Average Estimation of User Location Using the Most Common Value of *k* =1, 2, 3, … , 11 from Both the Noise Zone and the Stable Zone

Figure 5.8 shows the average of the correct percentages in the estimation of user location using the most common value of *k* =1-11 for both the noise zone and the stable zone. As mentioned earlier, in order to accept the result, the correctness shall be greater than 90%, for example, k=9 results in 90.4 % correctness and k=5 results in 92.7% correctness and when k greater than 4, the graph is always greater than 90%.

From the experiments, it can be concluded that:

1. the bigger the *k* number does not always mean the better the result (the best correct result)
2. *k* greater than 4 is recommended to allow a reasonable average of the best correct results to be reached.

5.8 Evaluation

The performance of machine learning in estimating symbolic user location has numerous variants; the result depends on the sensitivity of the area being examined. The algorithm can be tested in an extreme situation. For example, in estimating the location the notebook is placed next to a printer in a small area which is in conjunction with 3 corridors. The printer has a sleep mode and wakes up every 10 minutes for a few seconds when not in use. The interference in power from the printer changes the signal strength and signal quality and the result of the estimation of user location drops in correctness from 79% in the first hour to 34% in the next 14 hours.

In a normal situation, the testing of the estimation of user location reaches up to 96% correctness in the estimation of user location using WiFi's signal strength and signal quality for 14 hours. The estimation result may be improved in two ways:

- Improve the signal strength and signal quality by minimising noise in the building. This can be done by finding a better quality WiFi access point and installing uniform access points in the building.
- Improve the machine learning algorithm. Machine learning application in our implementation may be improved by adding more semantics, such as spatio (place), time and activity patterns, to the algorithm.

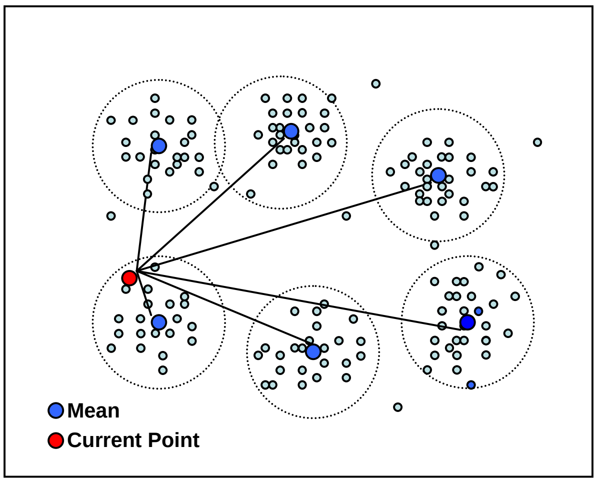


Figure 5.9 Normalisation of Signal Strength and Signal Quality Data Using Mean and Standard Deviation of Signal Strength and Signal Quality in the Room Scale.

From Section 5.5, it is obvious that *ηk*-Nearest Neighbour has a better result than *k*-Nearest Neighbour and that *ηk*-Nearest Neighbour(10) is the best algorithm among the four *k*-Nearest Neighbour algorithms, as shown in Table 5.4. On the other hand, it also shows that *ηk*-Nearest Neighbour(10) is *slower* than *ηk*-Nearest Neighbour. The *ηk*-Nearest Neighbour(10) produces an average result of 1.37 estimations per second and *ηk*-Nearest Neighbour produces 2.08 estimations per second, but the *ηk*-Nearest Neighbour(10) increases correctness by 5.5 % in estimating symbolic user location.

The performance of machine learning in estimating symbolic user location also depends on the training data set and the variety of the sensors. The wider the area covered by the use of a variety sensors (WiFi, Bluetooth, RFID, UWB, GSM/GPRS, and GPS) the more data collected. Bloggings technology and a spatio-temporal database location can be used to manage the fusing of such sensor data.

Table 5.9 The Difference (in dBm) Between Maximum Signal Strength in the Morning (08.50)

AP1	AP2	AP3	AP4	AP5	AP6	Room
1	4	3	0	0	0	Cor1
-3	4	4	0	12	2	Cor2
-4	-3	0	0	2	1	Cor3
1	-3	-13	0	-6	-5	E213
2	4	2	0	3	0	E214

Table 5.10 The Difference (in dBm) Between Minimum Signal Strength in the Early Evening (19.00)

AP1	AP2	AP3	AP4	AP5	AP6	Room
4	0	0	0	0	0	Cor1
18	0	0	0	0	0	Cor2
17	0	0	0	15	0	Cor3
33	0	0	0	0	0	E213
4	0	0	0	0	0	E214

As mentioned above, weakness of the signal strength is seen to fluctuate greatly in an indoor environment. Differences in signal strength can be analysed by measuring the signal at different times. In this experiment, signal strength was measured in the morning (08.50) and early evening (19.00) in springtime (Sept 2004). The measurement of the signal strength is in the room scale and calculates the difference between the maximum and minimum signal in the morning and the early afternoon. Table 5.9 for the maximum and Table 5.10 for the minimum show that the differences were quite significant. The maximum data of the signal strength in the room scale reached -13 dBm and for the minimum data reached -33 dBm in the signal range 0 to -99 dBm.

Based on this problem, the Boolean MaxMin algorithm can be used to evaluate the data set if it has good or poor quality in estimating user location. If the data was found to be of poor quality the next step would be normalisation of the data set. This can be done by calculating the mean and standard deviations in the room scale data. Figure 5.9 illustrates the normalisation process. Normalisation can significantly determine signal strength and signal quality which is applicable to a single location classifier, in this case a room.

The use of the Boolean Maxmin Algorithm to evaluate a training data-set before it will be used by the instance-based learning algorithm to estimate user location, is one example of the formal evaluation methodology, as described in Section 9.3. The purpose of the evaluation using the Boolean Maxmin Algorithm is to avoid a “snowball effect”[3] to the artefact (user location systems). This “snowball effect” can be avoided from the source of the problem investigated and solved from the beginning of the process, as early as possible.

In the case of the estimation of symbolic user location, which used *nk*-Nearest Neighbor, the main problem is how to avoid the possibility of multiple locations in the estimation of user location which will cause ambiguity. The training data-set needs to be converged to a single room or location. Thus, the training data set needs to be evaluated to avoid the occurrence of the “snowball effect” during the estimation process in the instance-based learning algorithm.

The Boolean MaxMin algorithm, as a formal evaluation algorithm, is designed to evaluate the training data-set whether it is a good quality data-set or not, i.e. the training data-set that converges only to a single room/location.

When the training data-set is evaluated, and the result shows that it is not a good quality training data-set, then the data-set can be normalised in an effort to convert it to a good quality training data-set. After the normalisation process, if ambiguous data still exists in the training data-set, the training data set cannot be used. The main problem in the estimation of user location is in the training data, which in this case, has no good determination.

The value of the lesson learnt from this case is that, the formal methodology of statistical analysis in machine learning can be used to evaluate the problem from the very beginning. It means that an objective evaluation can be achieved. However, without using the methodology, there is a high potential of getting the wrong location estimation, causing the weak estimation process, which can lead to the worst case scenario: wrong estimation.

When *nk*-Nearest Neighbor estimated correctly, the symbolic location algorithm is sufficient for many applications of location awareness, e.g.:

- As the basis of such user mobility patterns.
  Symbolic location history is useful when collected to form user mobility patterns by examining their semantic relationships with the current query in the local semantic region (Mantoro 2003; Mantoro and Johnson 2003; Mantoro and Johnson 2003), which can be used in future references. For example, it can be used for an automatic to-do list based on location information in the user location-aware schedule. When a user with his mobile device is close or matched to a certain location, his device can send an alert elegantly, to remind him of his task in his current location. More discussion of user mobility pattern based on user location is presented in Chapter 6.

[3] The **snowball effect** is present in any process that starts from an initial small state and builds upon itself, becoming larger perhaps potentially dangerous or disastrous (a "spiral of decline" or "vicious circle"), but might be beneficial (a virtuous circle, or often a self-fulfilling prophecy) (sources: http://encyclopedia.laborlawtalk.com).

- Instance group meeting based on location information.
  Exact location information can sometimes be helpful for a person in one situation but may be disturbing to others. Consider the scenario of a student struggling with home work and who knows precisely where other students are who are also taking the same course and who could provide help quickly, it would be possible to develop a meeting or study group by using videoconferencing which supports remote conferencing through video and audio streaming, such as GroupSys to support student group work which is described in (Alam, Walker et al. 2004).
- Being used as blurred location information.
  Sometimes a blurred location is useful for someone who wants other people to think that he is not where he is actually located. Consider the scenario in which someone leaves a home unoccupied for a time. The smart home can construct a model to present reasonable sequences of behaviour, which are executed in the owners' absence, giving the impression that they are at home. What is desirable in obscuring location for the home may be undesirable in the office environment: no single practice will fit all situations.

The accuracy of the symbolic user location in indoor environment (Active Office) is in room scale. If the external wireless provider is available in the same building, and the provider has more stable signals, which are in different channels (non-adjacent channel) and allow to be used, it may increase the accuracy of the estimation of the symbolic user location. Overlapping cells and frequency is no problem at all. However, the worse scenario, when the provider sets up his wireless signal in the same channel, a collision of signals will occurs and no wireless network will run.

In the use of this kind of quality location information in office or campus work, any mobile device with wireless capability can be used, as long as the type of wireless card used is the same class in receiving signal quality and signal strength. A user with this particular personal device will know his location and it may know other user locations and be possible to carry out group activity in an indoor environment.

5.9 Summary

This chapter focuses on the problem of estimating symbolic user location for closed (indoor) spaces using WiFi sensors. This work has three main strengths. First, real-life experiments and readings are provided to illustrate the fact that WiFi signals from access points can vary unpredictably, not only across perturbations in space, but also in time (diurnally). Second, this work proposes a simple but effective technique of normalising WiFi signals readings and using a *k*-Nearest Neighbour strategy to improve localisation accuracy. Third, as the quality of the training data-set plays an important role in estimating symbolic user location, this chapter proposes the Boolean MaxMin algorithm to evaluate the training data-set and prove whether or not the training data-set is good quality.

The WiFi has signal strength, signal quality and noise. The most significant of these three in the estimation of symbolic user location is signal strength, but the problem is that it tends to fluctuate greatly. This makes it difficult to correctly estimate symbolic user location. The solution to this problem proposes use of the machine learning algorithm using ηk-Nearest Neighbour in normalising the training data set and by combining it with 'the finding process of the maximum number of the locations that appear from the first nearest ten' of Euclidian distance. The normalisation of the training data set is an important process for two reasons: it develops the determinant for room scale and it increases correctness of estimating user location by 12.82%. The use of 'the finding the maximum number of locations that appear from the nearest ten' is also important because it also increases the correctness of location estimation by 3.4%.

Chapter 6

USER MOBILITY MODEL IN AN ACTIVE OFFICE

In an Active Office, user mobility has a direct implication for user activity, which is context-aware to the computing environment. Most often user location is the first variable to be considered when user activity is detected. Precise, proximate and predicted user location using a variety of sensors (e.g. WiFi and Bluetooth) was investigated in Chapter 4 and the algorithm using instanced-based learning to estimate user location was investigated in Chapter 5. This chapter discusses user mobility, which used the history of user location, by detecting the user's changing locations, and developing user mobility patterns. The user mobility pattern can be formed based on the number of visits and the use of time in a day's observation and also using a direct graph of user mobility. This pattern can be used for the prediction of future user movement based on *regularity*. The user's mobility pattern can continue to be improved by the *pattern of accuracy*, i.e., how sensitive is the accuracy of actual user mobility in the degree of regularity necessary to increase the level of the pattern of accuracy. This work is based on scalable context processing architecture for an Intelligent Environment as discussed in Chapter 3.

6.1 Introduction

Computers and communications systems today are underused because the range of control mechanisms and application interfaces are too diverse. It is necessary to consider the mechanism that might allow users to manipulate systems in simple and ubiquitous ways and to make computers more aware of the facilities in their surroundings (Weiser 1993; Harter and Hopper 1994). Context awareness mechanisms could be the best option to implement computer applications in an Intelligent Environment.

The Intelligent Environment is an environment with rapid and rich computing processing. The embedded sensing devices within the ubiquitous and ambient intelligent environment could be an important point in the Intelligent Environment, this is because the significant processing necessary to recognise user location and user mobility would be done by sensors. As a consequence, the environment with embedded sensing devices provides current information anywhere-anytime in the Active Office, which, covered by the sensors, then allows a user to access information on an anywhere-anytime basis as well.

This chapter discusses a known problem in ubiquitous computing: *user mobility* which is designed for an Active Office to allow a user to know his mobility patterns while allowing access to the local environment. The Active Office uses a variety of sensors/devices such as IEEE 802.11 or Bluetooth. to detect user location which then leads to user mobility.

The user mobility model discussed in this chapter is based on user's mobility patterns of the history of the predicted and proximate users' locations in an indoor space. The location history was collected using wired and wireless sensors and scalable context processing. This user mobility model has no relationship with user mobility in an

Immersive Virtual Environment and also has a loose relationship to user mobility in cellular phone wireless mobile networks, which enables a mobile user to communicate with others regardless of location. User mobility in this chapter is based on location awareness, as a part of context-aware mechanism which is aware of user location and all objects surrounding the user. The detail of the related work in user mobility presents in Section 2.10.

In a context-aware mechanism, the fundamental question of Who, Where, What and When for the Intelligent Environment is responded to by Identity, Activity, Location and Timestamp respectively. In this system the scope of 'Who' deals with user identity, persona, profile, personalisation and user model. The scope of 'Where' deals with user location. The scope of 'What' deals with user activity that can be represented by user mobility, i.e. the changing of a user's location to another location. Thus, a context awareness mechanism could deduce a user's activity based on the user's locations and time stamps where time stamp has responded to the "When" question.

These awareness-mechanisms in Active Office bring the computer into the user's daily activity and assist people with a variety of activities at work. A context-aware application has the capability to recognise user location, user mobility and social context where the social context is defined by the presence of other people.

6.2 What is an Active Office?

An Active Office is an implementation model of an Intelligent Environment. It uses scalable distribution context processing architecture to manage and respond to rapidly changing aggregation of sensor data (DiCPA architecture). The infrastructure supports interoperability of context sensors/widgets and applications on heterogeneous platforms. In order for an Active Office to provide services to users, an Active Office must be able to detect its current state/context and determine what actions to take based on the context, as also mentioned in (Dey, Abowd et al. 1999; Kummerfeld, Quigley et al. 2003).

User location is the most important part of user mobility because without understanding user location there will be no understanding of user mobility. In an Active Office, the terms of changing user location need to be clearly defined, since this work is not using coordinate mapping for the Active Office. The user is considered to be not *significantly moving* when he is typing on his computer, or opening the drawer of his desk. The user needs to change location significantly to be considered as having changed location. A "*significant*" movement means moving from room to room or moving from one side of the room to the centre or to the other side. The changing user location can be used to find the user moving from one location to another location and to explain user mobility. In an Active Office two important variables are considered, i.e.: *user speed* and *location resolution,* both variables that determine the quality of voice, video streaming, and email delivered to a user nomad moving from one location to another. This has an impact on the connection to a server such as an email server (IMAP/POP3) especially when the email carries multi-media files.

An Active Office is defined as a normal office, which consists of several normal rooms with minimal additional decoration (minimal intrusive detectors and sensors), and without explicitly badging people. An Active Office uses wireless communication, i.e.

Bluetooth and WiFi, to enable the understanding of user location, user mobility and user activity.

An Active Office environment should be simple, efficient, scalable, fault tolerant and applicable to a range of heterogenous computing platforms.

6.3 Hotspots and User Mobility

A user mobility application can be the basis of a future wireless system, especially in an Active Office, by providing multiple overlay networks, which have different characteristics. The networks require support for the seamless delivery of popular desktop services such as web services, video streaming, email (IMAP) or even updating a meeting schedule in the user mobility device. This support can increase the nomadic lifestyle of the user (Chan and Seneviratne 1999).

In understanding the pattern of user mobility, user nomadicity in an Active Office can be studied as a user moves from one hotspot to another hotspot, or in the intersection of two or more hotspot areas. The possibility of a user moving in the WiFi's hotspot areas in an indoor environment needs to be investigated. Most often the hotspots have intersection between rooms. Figure 6.1 shows the possible movements of a user in a hotspot area that may cover several rooms.

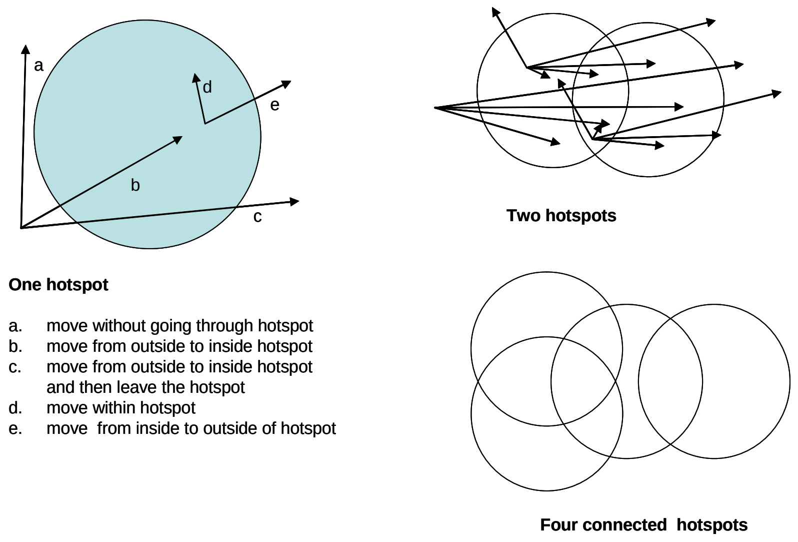

a
d
e
b
c
Two hotspots
One hotspot
a. move without going through hotspot
b. move from outside to inside hotspot
c. move from outside to inside hotspot and then leave the hotspot
d. move within hotspot
e. move from inside to outside of hotspot
Four connected hotspots

Figure 6.1. The User's Possible Movements in the WiFi's Hotspot Areas

Consider two different points of user location: inside and outside a hotspot area. The possibilities for the user to move from outside the hotspot include:

- moving into the hotspot area
- moving into the hotspot area and moving back out of the hotspot
- moving around the Active Office without going through the hotspot

The possibilities for the user to move from inside of the hotspot include:

- moving around inside the hotspot
- moving out of the hotspot area

When the number of hotspots is increasing, the impact may be that the number of intersections also increases. As a consequence, a context-aware application to handle user mobility also becomes more complex since the possible number of user movements is also increasing. This complexity brings other problems which need to be solved by a context-aware application, such as overlapping frequencies, channelling and degradation noise, with the user roaming seamlessly between cells and experiencing differing network performance.

6.4 The Active Office Area of Study

Every office within an organisation has its own characteristics in the management of the office's activities, such as finance/budgeting, human resources, computing resources, even though the offices all exist within the same organisation. These characteristics have direct impact on the design of an Active Office and complex design can appear to be the result when it is implemented for the office's daily activities.

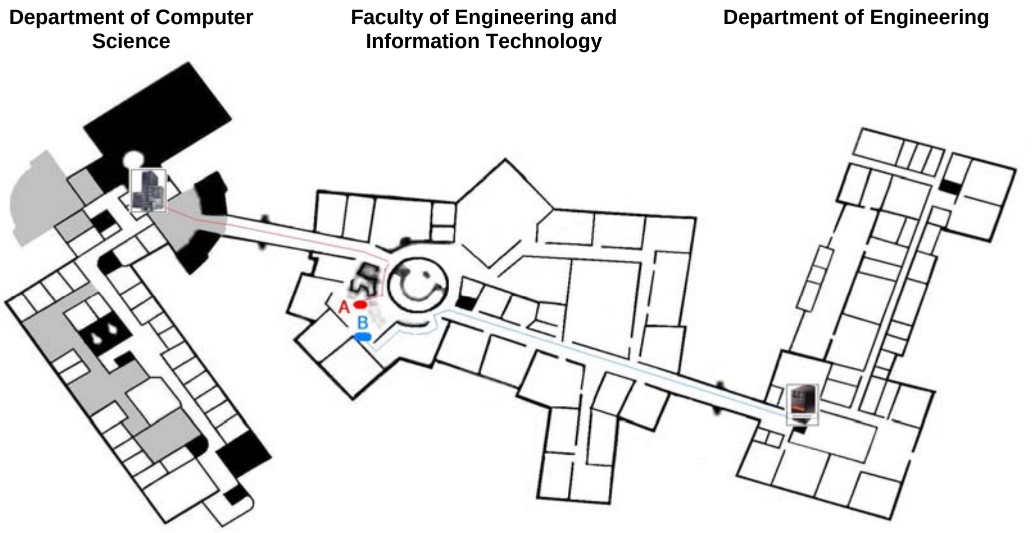

Department of Computer Science
Faculty of Engineering and Information Technology
Department of Engineering
A
B

Figure 6.2. Three Buildings Housing the Faculty of Engineering and Information Technology as an Area of Study of the Active Office.

For example, one characteristic of the management of office activity in The Australian National University (ANU) is that each of its offices has its own budget to be managed each year, including hardware and software network budgets. The Faculty of Engineering and Information Technology (FEIT) – ANU has 3 such offices shown in Figure 6.2. Each office is in its own building, i.e., Department of Engineering (3 floors, 81 rooms), Faculty Office (2 floors, 50 rooms) and Department of Computer Sciences (3 floors, 90 rooms). The three buildings are connected to each other at the second floor level. Each of the buildings has wired and wireless network connection, so that, when a user moves along the second floor corridor from one office to another, the user will always have connection to the network. There is a problem, however, in that when a user who belongs to the Department of Engineering walks through to the Department of Computer Science and downloads a large file, the charges will bill to the Department of Computer Science, which is not appropriate. Another problem is that each building has its own servers and most of the servers have different operating system platforms i.e, Windows 2000, RedHat/Fedora, and Solaris. Each office is also managed by different system administrators.

From the problem examples above, it is obvious that a context aware network application is needed to allow a user to connect to the network in his current location as well as when he moves (changing locations) from an area in one building to an area in another building, but one by which the incoming and outgoing data should access and calculate the budget in the local server of the user's base office.

To develop this application, several public domain software applications were deployed, such as Winbind, VPN (Virtual Private Network), DHCP (Dinamic Host Configuration Protocol), NAT (Network Address Translation), and LDAP (Lightweight Directory Access Protocol). Security is one of the crucial aspects to be considered for this purpose. The software applications below were implemented:

- Winbind is a module to map windows NT domain databases to Unix and is the lastest development of the Samba application in Unix.
- VPN is implemented using PoPToP/PPPd and CIPE (Crypto IP Encapsulation) which works by tunnelling IP packets in encrypted UDP packets
- Login id and password uses LADP from university-wide services
- NAT to provide firewall using iptables (IP Masquerading)
- DHCP is setup in two different modes: non-authoritative for wireless users by registering all users' MAC address wireless devices and authoritative for wired user devices (laptop/notebook, PDA, Smart Phone, PDA Phone)

Points A and B, in Figure 6.2, illustrate person A, who belongs to the Department of Engineering, meeting person B, who belongs to the Department of Engineering, in a room in the Faculty Office. When they have connection to the network in the Faculty Office, their incoming and outgoing data do not go to the Faculty Office but to each department as appropriate.

Further context-aware applications need to provide user-facility not only to connect to the network but also to access the variety of general public resources in the local network, such as burning cd/dvds and printing in another building, while the budget calculation is in the user's base office. The traditional approach is to access the variety of general public resources in applications by imposing uniformity on the environments. This approach is not adequate. The described alternative architectural distributed processing framework which was discussed in Chapter 3 is better matched to the needs of ubiquitous computing, especially when the configuration changes or the resources come and go. The computing environment is designed to have capability to adapt appropriately.

A context-aware application for user mobility purposes has the capacity to connect at different points of attachment in other Intelligent Environment domains to run context-aware applications to access resources and services while the user is in motion. The case example is a user working on his PDA phone that can change its network interconnection transparently with regards to the applications. When the user is in motion from one location to another, a context-aware application has the capability to maintain all previously established connections and make them usable at the new location.

6.5 The Pattern of User Mobility Based on History Data

This section discusses user mobility in an Active Office, where the user changes from one location to another location. The user location data is collected and managed as describe in Chapter 5, including the algorithm and the design of the databases. This area in ubiquitous computing is called location management or location tracking and incorporates the set of mechanisms with which the system can locate particular user mobility at any given time. Two strategies are available: location update and location prediction.

Location updating is a passive strategy in which the system periodically records the current location of the mobile users in a database that it maintains. Tracking efficiency is based on the frequency of updates that are initiated by the user’s mobile devices.

Location prediction is a dynamic strategy in which the system proactively estimates the mobile’s location based on a user-movement model. Tracking capability depends on the accuracy of the model and the efficiency of the prediction algorithm.

As described in Section 8.2.2.1, location estimation is different from location prediction. Location estimation is the use of proximate sensor data using machine learning algorithm to estimate a user location. Location prediction is the use of probabilistic method to predict user location based on patterns of historical data of fixed sensors and proximate sensors data.

Table 6.1. History Data Summary of a User’s Mobility for One Day

Room No.	Duration (Second)	Duration (9am-5pm)	Room visit (times)	Duration (%)
DCScafe	1868	0.518889	5	7.412698
N314	2177	0.604722	5	8.638889
N323	2692	0.747778	6	10.68254
N326	780	0.216667	1	3.095238
N329	1721	0.478056	1	6.829365
N330	3148	0.874444	4	12.49206
Reading room	1324	0.367778	1	5.253968
Resources room	1324	0.367778	1	5.253968
Seminar room	1663	0.461944	3	6.599206
Stairlevel1	1088	0.302222	3	4.317460
Stairlevel2	2207	0.613056	7	8.757937
Wedge/Corridor	2339	0.649722	6	9.281746
Working groups room	2869	0.796944	6	11.38492
	25200		49	100%

The motivation of location prediction is simple, when precise sensors and proximate sensors data is not available, user may turn off his service location and the system needs to supply user location information, location prediction is the only close approach to know user location. So, when precise location from precise sensors and location estimation from proximate sensors is not available, the location prediction using history data will predict user location based on user mobility pattern.

Location management is generally perceived as purely a database updating and query procedure from a spatio-temporal database. This is quite different from location management in a cellular wireless phone network, which requires paging cells. The new location area consists of a number of cells with the exact location of the user device while in motion determined for call delivery in paging the cell in the last registered location area. The primary goal of the location update is to reduce paging costs (Cayirci and Akyildiz 2002). Paging only occurs when the phone is not making a call. During a call it is tracked. Paging usually only occurs during call set up.

As also described in Chapter 5, the accuracy in estimation of user location reached 96% within 14 hours for every single user; the task of locating the user given his last location would be substantially efficient considering the speed and the system resources used while the user was in motion.

The representation of user mobility patterns can be created by calculating a summary of the user-location history and then generating a line chart of the number of users visiting rooms, the number of users spending time in each room and a direct graph of user mobility.

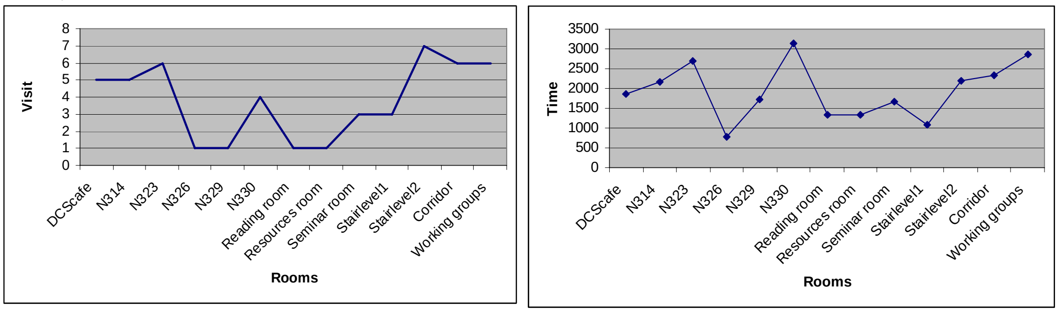


Figure 6.3. The Pattern of User Mobility Based on the Number of Rooms Visited and Time Spent (in Seconds).

Two voluntary users are observed in the study of user mobility. The first user has a room on the third floor of the DCS building, where his mobility pattern will be observed using a line chart, while the second user has a room on the second floor of the same building, his mobility and his activity will be observed using a graph approach. Table 6.1 shows the summary of user mobility data for one day. This table summarises actual sensors data in the office of the first user, which recorded up to 25200 seconds. The first user was very active on that day changing location 49 times in room scale. The table does not show his activity. Figure 6.3 shows the pattern of user mobility relating to the number of rooms visited and time spent within a 7-hour observation period.

The summary of the second user's observation data, including possible activities, is shown in Table 6.2. This data summary is also based on user location data history. The user activity in this table is recognised in the reports of the available sensors, such as keyboard or mouse sensor or a note taken manually when he visit rooms, in the corridor or on the stairs. The technology to recognise user activity in an Active Office was still in the early stages of development when this experiment was performed.

The user arrived at his office about 9am and left the office about 5pm, he was undetectable for about one hour during his lunchtime outside the office, he possibly had lunch.

Table 6.2. User Mobility Sample Data with Activities in One Day

Room	Activities	Time	Duration (minutes)
Stair Level 1	Walk through	08.58-08.59	1
Corridor	Walk through	08.59-09.00	1
N235	Work on the desktop computer (work with email, programming) Read papers Makes notes	09.00-10.12	72
Toilet	Using the toilet	10.12-10.16	4
N235	Work on the desktop computer (work with email, programming) Reading email	10.16-11.02	46
Corridor	Walk through	11.02-11.03	1
DCS café	Morning tea	11.03-11.19	16
Corridor	Walk through	11.19-11.20	1
Resources Room	Pick up printing Check email	11.20-11.23	3
Corridor	Walk through	11.23-11.24	1
N235	Reading a Paper Making a note	11.24-12.31	67
Corridor	Walk through	12.31-12.31	1
Stair Level 1	Walk through	12.31-12.32	1
Undetectable	Out for lunch	12.32-13.28	56
Stair Level 1	Walk through	13.28-13.29	1
Corridor	Walk through	13.29-13.30	1
N235	Work on the desktop computer (work with email, write a paper) Read papers Make a note	13.30-15.35	125
Toilet	Using the toilet	15.35-15.38	3
N235	Work on the desktop computer (work with email, write paper) Write a paper	15.38-15.55	17
Corridor	Walk through	15.55-15.56	1
Stair Level 1	Walk through	15.56-15.57	1
Seminar	Join DCS Seminar	15.57-16.50	53
Stair Level 1	Walk through	16.50-16.51	1
Corridor	Walk through	16.51-16.52	1
N235	Work on the desktop computer (check email) Clean up tables	16.52-17.02	10
Corridor	Walk through	17.02-17.03	1
Stair Level 1	Walk through	17.03-17.04	1
Undetectable	Not reported	17.04 - …...	

The direct graph at Figure 6.4 shows the pattern of user mobility, how the user gets in to the office and moves around his office. The vertices represent rooms that the user visits with the edge representing user movement from one location to another. All user movements in the Active Office are recorded 3-4 times continuously using WiFi, and other sensors (Bluetooth, keyboard activity sensors, mouse activity sensors and pressure sensors) that recognise possible user locations and user activities. All possible activities are also recorded including toileting.

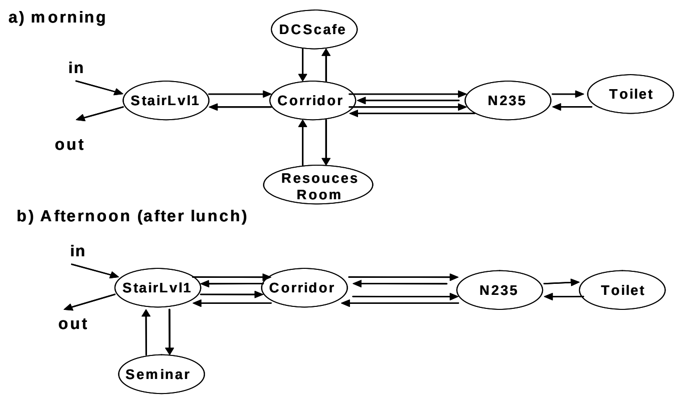


Figure 6.4. Pattern of User Mobility using Direct Graph in the Active Office.

In the design of a user interface for tracking user location, the user has the capability to turn off their location tracking system. This capability is built in for the purpose of user privacy. For example, toileting may indicate that the user location system tracks the user and broadcasts the fact that user is in the toilet. Some users who may not wish to be tracked for privacy reasons, they can turn off their location tracking system. However, the tracking individual users to the toilet has several advantages, for example, it can be used to monitor how many times a user goes to toilet for user health purposes and also it can be used to show the toilet location as part of the building plan for security purpose.

By collecting patterns of user mobility from histories of user location over a long period of time, patterns allow significant understanding of user behaviour in the Active Office, such as:

- the most probable user location
- measurement of productivity

User productivity measurement may be based on how long the user works with his computer within a given week, how many guests visited him, depending upon the user's type of work. This could lead not only to greater user productivity in the Active Office, but also has implications for other activities, such as work distribution, that relate to the office work cycle, such as high end activity at the beginning or end of the financial year and before Christmas. High end activity can be recognised by the time spent by the user in his office, especially after office hours, outside his normal time schedule.

In an Active Office once a user's location is captured patterns of user mobility can be mapped. The dynamic map of user mobility shows the *regularity* of user activity, which can be used in the prediction of user activity. The user's mobility pattern (map) can be improved by increasing the level of *pattern accuracy* by adjusting the degree of the regularity of user mobility to actual user mobility. Regularity is the probability of user mobility in following the user's daily habits. Regularity basically monitors user mobility and follows the user's regular movements in the Active Office.

The pattern of user mobility may be used for future reference, for example when a user is found to be sick, the pattern of the user's mobility for several days before he is sick can be studied. Once a mobility pattern is created an early warning system can be created to warn the user when is the time for him to sit back and relax or even to take a break. The pattern can be created as a dynamic pattern by validating and using the regularity and pattern of accuracy concept.

The user mobility pattern can also be used to predict what a user will do for a short period. A possible future application, for example, is a reminder for a user not to drink too much several hours before an important meeting.

Other possible future applications using this method could see the implementation of plans for the user's dietary, body training, and sleep behaviour needs and require the user to take his small device (smart phone/PDA phone) everywhere with him.

This could be a contentious proposition and see disagreement with the idea of being monitored and nagged by personal devices.

6.6 Summary

This chapter proposed a user mobility model in an Active Office based on significant changes to user location. In an Active Office the users have regular work schedules and routine activities at specific times. Based on the history of the user's location, user mobility patterns can be developed and can be used to predict the user's location and his future activity in a specific time frame.

A user's activity can also be represented by user mobility, which can be seen from the user's changing location. Thus, in an Active Office, the collection of user's locations represent user mobility and the collection of user mobility can be mapped as a pattern of user mobility. The dynamic map of user mobility shows the *regularity* of user activity for the prediction of user mobility. The user's mobility pattern (map) can be improved by increasing the level of the *pattern of accuracy* by adjusting the degree of the regularity of user mobility to actual user activity.

Chapter 7

USER ACTIVITY BASED ON LOCATION IN A DISTRIBUTED CONTEXT-AWARENESS ENVIRONMENT

The capability of a computing environment to capture user activity is very limited, while the computer environment itself changes over time, making user activity the most complex part of the context awareness mechanism in an Intelligent Environment. The problem for the user is how to enable transparent distributed computing to continue operation across changing circumstances in a seamless manner while the environment recognises that a particular user activity is different from others which can be in serial or parallel activity.

This chapter describes an approach to the determination and recognition of user activity based on user location. Human activity is extremely complex (Crowley 2003) and determining user activity is not simple. Some activities are continuations of previous ones and sometimes one activity depends on another activity which belongs to another user. One such unique situation is the activity that occurs when a user has a guest in an Active Office and how the Active Office deals with an "unregistered" user or unknown device in the social model and how a guest user, a user without enough knowledge of the local network, gains access to the Active Office. User activity of a guest depends on the host who provides access for the guest. A guest user could have limited access to the resources of an Active Office, which is a non-familiar domain for the user. For this purpose, a mobile access point for guest users was introduced. This work is based on DiCPA architecture, a distributed context processing architecture for an Intelligent Environment as discussed in Chapter 3. The characterisation of user activities in an Intelligent Environment, which can be seen from sensor activity on the scalable distribution of context and context information, is also discussed.

7.1 Introduction

Scientists in the Intelligent Environment area are researching ways to make embedded computing and Ubiquitous Computing work better for people by creating and equipping an Intelligent Environment, such as an active home or an active office, with technologies that can identify the user's needs and meet them speedily, efficiently and unobtrusively.

The goal of Context-Aware Computing in general is to make user interaction with the computer easier in the Intelligent Environment where technology is spread throughout (pervasive), computers are everywhere at the same time (ubiquitous) and technology is embedded (ambient) in that environment. A context-aware application should reduce the load of the user and adapt to them seamlessly (Kern, Schiele et al. 2003; Lukowicz, Ward et al. 2004). While a user is doing his daily activities, his access to the Intelligent Environment should not be difficult, tedious or need considerable learning on the part of the user. The interaction should be safe, easy, simple and enable new functionality without need to learn a new technology. As human activity is a central part of the user context (Kern, Schiele et al. 2003), the context-aware system would provide relevant information and a simple way for a user to deal with the computing environment. Context information cannot be supplied by the user. It should be sensed automatically using sensors in the computing environment, in making these smart environments have the capability to assist and help people with a variety of activities by detecting a users' current state/context to determine what actions to take based on that context.

In an active home, the benefit of such technology might simply be convenience in the home, for example, knowing when the occupant wakes up and what radio station they like to listen to without waking up the rest of the house. On the other hand the active home could be life saving by detecting when an occupant has fallen and needs medical help.

So how can technology identify exactly where a person is in an environment, so that the environment can anticipate and meet the user's needs, perhaps even before the user knows what those needs are? Consider a system that can identify when users are getting too hot, and which can automatically adjust the temperature of a room to a cooler temperature. However, this system will behave differently if a user is exercising (by blowing air at the user) than if a user is at his desk working (by cooling the air in the whole room, without strong breezes to blow papers around). The environment should not waste energy heating or cooling empty rooms. This calls for a smart system that not only knows where the user is, but what he is doing.

This approach to locate a person within an environment uses the wireless connections in devices that are normally carried for other purposes, for example, a mobile phone, PDA or a laptop computer. The location of these devices, and hence the person with them, is determined by a mixture of precise, proximate and predicted location sensors. The data from these sensors is turned into a predictor to precisely locate the device, and thus the person. Once a user is located such services can be delivered based on the current situation from a resources manager.

User activity can be shown from sensor activity in capturing changes in state, time and location. To manage and respond to rapidly changing aggregated sensor data, DiCPA architecture is used, as discussed in Chapter 3. This scalable distribution context processing architecture in the Intelligent Environment allows continued operation across changing circumstances for users, the collection of nearby people and objects, accessible devices and changes to those objects over time in the environment. The DiCPA architecture is used in the implementation of user location and user activity model. The understanding of user location and user mobility leads to the understanding of the concept of user activity. The context information of this user activity can be used to characterise the user situation.

7.2 User Activity Concept

In the Context-Aware Computing or pervasive computing discipline, user activity is not a well defined concept, this may be because of:

- the transition between activities is sometimes not very clear,
- an activity can be seen as part of another activity,
- an activity can be in parallel mode, in sequence mode or both together,
- different views can be deduced from different activities.

For example, the difference between running and walking activity; it needs clear variables to be defined, several variables which may involve, e.g., the length of the user's leg, the speed of movement or the number of steps taken in each minute. When a user cleans his house, he needs to walk or move. Thus, it may not be clear whether the activity of the user is walking or cleaning the house. A different view and a different perspective can lead to the assumption of a different user activity.

User activity in the Intelligent Environment can be divided into 4 categories:

a. activity as association between a user and smart sensors in the environment
b. activity as a node in a work flow or job breakdown
c. activity as a physical movement mode or state
d. activity as a mode of state of human intent

In this study, user activity is defined as *any association between a user and smart sensors in the environment* or *any sensors being in active use to access the resources*. Smart sensor can be one or more sensors (array sensors) with the integrated application that has the capability to make decisions for certain purposes. Aggregation of sensor data is only one of the processes in the sensor application to characterise user activity.

To recognise a user's activity while accessing resources, a context-aware application requires user identification. A user's identity can be captured from the user's mobile computing devices or user's image/voice recognition. Users can be characterised by several means, i.e., identification and authentification, user profile, user's terminal and user's access network characteristics, and service adaptation to user environment.

User characteristics can be recognised when any association exists between a user and smart sensors, and the association is recorded in a sensor database which contains information relating to user identity, sensor identity, location identity, time and state. The collection of this data, user activity history data, can be used to form a pattern of the user's mobility, as discussed in Chapter 6, with the regularity of a user's activity pattern also capable of being recognised. This pattern can be used to predict a future user activity based on a user routine activity by querying the user activity pattern.

Many earlier projects acknowledge the need for the capability to capture user activity in the complex relationships between a user and the environment:

a. the activity as an essential ingredient for determining appropriate reactive behaviour as requirement for Response Offices at Xerox PARC (Elrod, Hall et al. 1993)
b. The need for *tracking activity* such as EasyLiving at Microsoft acknowledges (Brumit, Meyer et al. 2000), or *tracking activity of daily living* to identify the short- and long-term changes in the health of elderly people (Patterson, Fox et al. 2003).
c. the utilisation of the *activity based approach* such as in the second generation of iRoom at MIT (Kulkarni 2002). This approach is similar to the *activity-centric context model* at DSTO, which have agents and activities as key components. The activity centric context model adopts a knowledge management approach to support the hand-over of artefacts from one individual or group to another. For example, to hand over a system design it is necessary to pass the formal artefacts of the design activity but it is also important to pass the tacit system design context, the mental models, assumptions and other factors that exist when the design was developed (Prekop and Burnett 2002).

In this work, the capability to capture user activities outlined above was considered and followed the prediction of future user activity by the use of history data to form an activity pattern.

7.3 Activity-based Processing Model

An activity based approach uses the abstraction of the sensor data that is accessed by a user in the computing environment. In this part, the activity-based processing model will be discussed. The model has 5 stages as shown in Figure 7.1, i.e. Sensors, Smart Sensor, Resolver, Resources Manager, and Presentation.

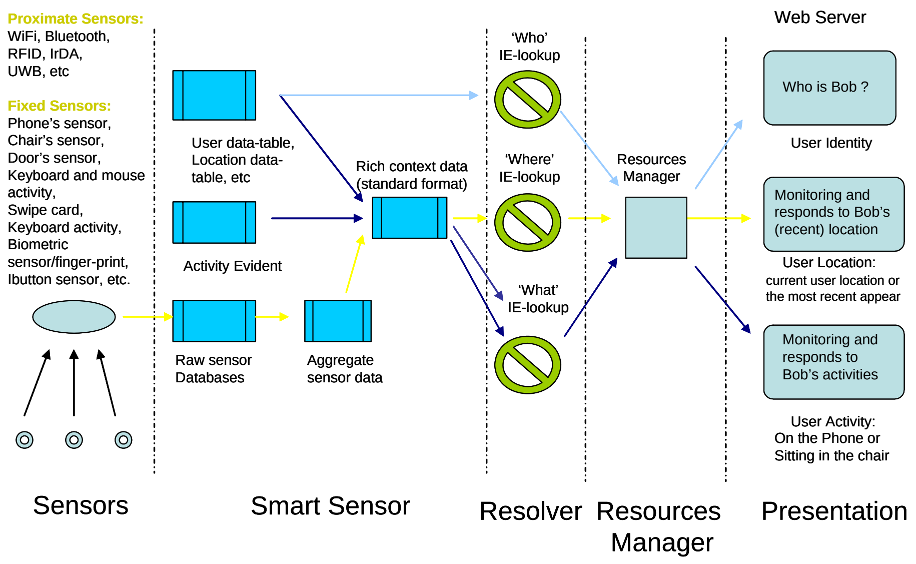

Proximate Sensors:
WiFi, Bluetooth, RFID, IrDA, UWB, etc
Fixed Sensors:
Phone's sensor, Chair's sensor, Door's sensor, Keyboard and mouse activity, Swipe card, Keyboard activity, Biometric sensor/finger-print, Ibutton sensor, etc.
User data-table, Location data-table, etc
Activity Evident
Raw sensor Databases
Aggregate sensor data
Rich context data (standard format)
'Who' IE-lookup
'Where' IE-lookup
'What' IE-lookup
Resources Manager
Web Server
Who is Bob ?
User Identity
Monitoring and responds to Bob's (recent) location
User Location: current user location or the most recent appear
Monitoring and responds to Bob's activities
User Activity: On the Phone or Sitting in the chair
Sensors
Smart Sensor
Resolver
Resources Manager
Presentation

Figure 7.1 User Activity Processing Model

7.3.1 Sensors

In recognising user activity, the principal questions are: How many sensors are needed to recognise a user activity and at what precision? Can activities be recognised using simple sensors that detect changes to the states of objects and devices in a complex office setting? For a simple activity, a single simple sensor can often provide powerful clues to activity. For instance, a keyboard activity sensor can capture user typing activity and a pressure chair sensor can strongly suggest that a user is sitting on the chair, both types of sensors can show user location as well. However, these sensors cannot show other activity, such as that a user has a meeting activity. In essence, there is no one answer to the question posed above; the number of sensors needed to recognise user activity depends on the type, function and precision capability of the sensors used to capture the user data

As mentioned earlier in Chapter 4, an Active Office has two types of sensors, i.e., proximate and fixed sensors. It uses proximate sensors, such as WiFi, Bluetooth, RFID and IrDA. These sensors have been used to sense user activity. The other sensors can be added as required by the room in an Active Office to capture user activity, such as UWB, or eye-movement sensors. For fixed sensors, magnetic phone sensors, pressure chair sensors, magnetic door sensors, keyboard activity sensors, mouse activity sensors and swipe cards are used. Fixed sensors can also be extended to other sensors such as biometric/finger print sensor, iButton sensor.

In this model, both sensors provide the raw data recorded in spatio-temporal databases, as discussed in Chapter 8.

7.3.2 Smart Sensor

The smart sensor in this model is based on three kinds of data, i.e., raw sensor data, activity-evident data, and data-tables, such as user data-table, location data-table. The raw sensor data which is recorded in a spatio-temporal database is a key entity for the smart sensor to deduce what kind of user context information is needed in any situation. Aggregate sensor data is the extracted data from a raw sensor database. This type of database can be used for two purposes, i.e.: speeding the process query and showing the scalability of the query.

7.3.3 Resolver

Resolver is the procedure for looking-up user identity, location and activity. This approach is similar to the DNS server lookup host table. DNS server can resolve host name to ip-address and vice versa. In this model, the resolver uses the DNS idea more widely, it resolves three variables i.e., User Id, Device Id and MAC address. It is possible for a user to have several devices and possible for each device to have several MAC addresses, and hence possible for a user to have many identities in the environment. There are three functions of the resolver in this work that have been designed for Active Office purposes, i.e. for user identity purposes, it uses user identity lookup tables; for location purposes, it uses scalability of location lookup tables and for user activity purposes, it uses several entities (databases) to deduce user activity.

7.3.4 Resource Manager

In Active Office's network management, a resource manager acts as the coordinator. It maps available resources. It contains agents such as resolution agent, inter-domain agent, ICMP agent, SNMP agent, Content Routing agent. For user activity purposes, some of the resources manager's functions may not be used much, such as accepting the object's global name from the resolution server or the use as persistent mapping to request persistent location from the resolution server. In this part, the resource manager's function is to coordinate the resources based on the status of the sensor data including the aggregated sensor data to provide a complete set of context-aware information which contains user identity, location and activity information.

7.3.5 Presentation

The presentation, response or action is based on data processing from the resource manager. User activity can be shown whether virtually, in a web page, in computer monitoring, or in direct action to the user.

7.4 The Role of Location to User Activity

An important problem in an Intelligent Environment is how the system can characterise user situation based on user activity, where user activity is based on any objects (such as smart sensors) being in action relating to a particular person's use. It could provide complex and rich information which is relevant to the situation/domain being examined.

This information is used to characterise the situation of user entity and environment entity, where user entity is a person being in the environment and an environment entity is an indoor or outdoor space that is embedded/equipped with smart sensors. A complete set of user activities can be identified by reading and interpreting any association between a user (user entity) and smart sensors (in the environment entity).

When smart sensors sense a user and recognise a user location, the possible activity of the user can be estimated based on the user location itself. For example, in Table 7.1, when a user is found in his office (N235), the possible activities in his room are working on the computer, working on his table, using the telephone, or having a meeting. Table 7.1 is the summary of Table 6.1 in Chapter 6. In Chapter 6 user activity is reported by the user while he is on the move. In this part, how to recognise user activity based on sensor data will be discussed. This approach does not need the subject to label his activities such as in (Tapia, Intille et al. 2004), where it could be difficult to adapt to individual patterns of activities.

The activity can be structured such as in Figure 7.2, this will clarify the tree of the user activity. When a user is working on the computer, the mouse and keyboard activity sensors can be used to detect in which windows and with which applications the user is engaged. Hence, when the location sensor found his location is in room N235, the mouse and keyboard sensor can then report an activity such as user id, windows and application. These could then be used to deduce the type of user activity.

In the case of a user working with email, there are many ways to access email, such as remote access and open mail agent, create a pop email script and convert to special word format. This is not recommended in the Active Office, because it will lead to difficulties in recognising user activity. In the Active Office, working with email is only recognised when a user makes common use of a regular mail agent (such as Microsoft Outlook, Mozilla, Netscape, Eudora, Pine) or web-mail (such as mail.yahoo.com, www.gmail.com).

Recognising user activity when the user may be working on his table is not simple, so far in the Active Office only user location data have been recognised from WiFi signals, and chair sensor data using a pressure sensor that is embedded in the chair (see Chapter 4), but as yet no sensor has been embedded in the table. Once the Active Office has a pressure sensor covering the whole table, the Active Office will be able to deduce when the user is working on his table. However, the situation where a user works on his table is a different situation from that where the user is on the phone, since, as the Active Office has a phone sensor, it can be easily recognised when the user is having a phone call.

In the case where the user has a meeting, when it is found that other users are also in the same location (room) for a certain period, it can be deduced that the user has a meeting with other people. If the other users status is recognised as student then it is a meeting with a student, the same thing can happen when the user identity is found to be his supervisor. A context-aware application will deduce the activity based on the recognition of the user’s identity. However, when there is a user/guest and his location is recognised but no identity is available, then the Active Office needs to find a way to recognise the existence of that user/guest, one option is by identifying his mobile device using resolver as discussed on Section 7.3.3.

When a user leaves his office, user activity status will be undetectable. This status will be the same when the user does not allow the Active Office to detect his location. When a user’s location is found to be in a seminar room, and it is also matches with the seminar schedule, it can be deduced that joining the seminar is his user activity status.

- Working on the computer
  - Office Application
    - Word processing
    - Spreadsheet
    - Presentation software
  - Mail agent
    - Read email
    - Forward and write email
    - New email and write email
  - Web application
    - Download paper/image
    - Searching information
    - Web email
      - Read email
      - Forward and write email
      - New email and write email
    - Monitoring equipment
- Working on the table
  - Read a paper
  - Make a note or writing a paper
- On the phone
  - Internal call
  - Local call
  - Inter-local call
  - International call
- Meeting
  - Meeting with other staff
  - Have a guest
  - Meeting with supervisor

Figure 7.2 Example of Tree Structure of User Activity

A similar deduction may be made when a user is found in the DCS café; if he has stayed for more than 5 minutes and the time is in the morning about 11am, then it can be deduced that he has a morning tea activity. This applies also for afternoon tea activity.

When a user's location is found to be in a corridor or on the stairs, and his location changes at a reasonable speed, it can be deduced that he walks in a corridor/on stair activity. When a user's location is found in toilet for more than 3 minutes, it also can deduce that he may be so engaged.

Table 7.1 shows the summary of the duration of a staff member's activities on a certain day, sorted by the duration of visits to rooms. This also shows that the user spent 69% of his office hours on that day in his office. This approach proposes the tagging of each room with the function of the room and how often the users have activities in a particular room.

To have a better understanding of the recognition of user activity based on location, activity zones can be created, as explained in (Koile, Toolman et al. 2003). Activity zones can be mapped by including observed location features in regions corresponding to activities or sets of activities. In this study, a 3-D pattern of user locations is not further explored, instead short-range sensors, such as an RFID sensor, which can be embedded in any Active Office object is used to allow recognition of user activity. 3-D location of users or an RFID sensor is useful but alone is insufficient as context information. In this study the raw RFID sensor data is combined with raw WiFi data.

Table 7.1 Summary of a Staff Member's Activities on a Certain Day.

Room	Visit (times)	Duration (minutes)	Possible Activities
N235	7	337	Working on the Computer Reading a paper Make a note or writing a paper Telephone Meeting
Undetectable	1	56	Out of office Not allowed to detect
Seminar	1	53	Join DCS Seminar
DCS café	1	16	Afternoon Tea Morning Tea
Corridor	9	9	Walk through
Toilet	2	7	Toileting
Stair Level 1	6	6	Walk through
Resources Room	1	3	Picking up print out *Check mail* Fax Binding Pick up stationery/paper Pick up reading material Photocopy

The partition space/zone is based on simple proximity or relies on user specified maps of regions. Location regions can be learned from observed activity, including user changes to location/motion. Each zone corresponds to a region in which a person is likely to be engaged in similar activities. Activity zones can overlap in space, since motion can indicate a different activity. This activity map can be used at run time to contextualise user preferences, such as allowing "location-notification" settings of messaging, environmental control and multi-media delivery (Koile, Toolman et al. 2003).

For example in the case of a resources room in the department of Computer Science, ANU, as shown in Figure 7.2, if a user's location is found by WiFi sensors to be in the resources room, and RFID sensors in front of the Reading Material zone sense that a book has just been moved from the shelf, the Active Office can deduce that the user is picking up a book.

A more complex application allows the recognition of a complex user activity by the addition of the semantics of an object (knowledge of objects such as chair, desk, and computer) and the semantics of human behaviour (such as people come to the office around 9 am, read and write at a desk). In an Active Office, user activity depends on the type of organisation and the location/position. In the case of the University organisation, especially in the head of department room at department level, the most frequent activities are working on the computer, talking on the phone, having a guest, a staff meeting, or meeting with a student. According to Stinson, the use-of-time can be placed into two categories; *in places* and *in transit* (Stinson 1999), the details have been explained in Section 2.11. Any user activities in Active Office are *in places* activities. Table 7.2 shows the list of possible activities *in places*, based on observation of location (room) function in a University organisation. This list can be added to, but the list in Table 7.2 is the representative of the major activities based on room type. For example, the most possible activities in a Head of Department room are; working on computer, talking on telephone, having a guest, staff or student meeting.

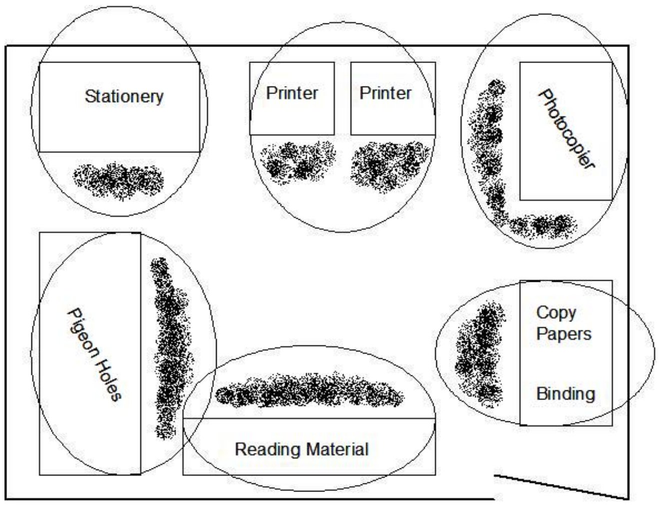


Figure 7.3 Access Zone in the Resources Room

Some activities in the Active Office, such as in the University, could easily be detected, for example working on the computer or talking on the phone. However, recognising the difference between when the Head of Department is having a guest, the Head of Department is meeting with a staff member or a student, is a still problem, since at the moment not all people have an identity that can be recognised by the Active Office. If user productivity is a goal of the Active Office, it can be measured by counting the number of tasks completed per unit of time, and can also convert these measurements to measurements of time per task (Nielsen and Levy 1994).

The most complex activity occurs when a group of people share their activities. The social exchange and actions within a group of people have direct impact on group activity. A working group is situated in a rich context of organizational strategies and objectives, job roles and responsibilities, interpersonal relationships, task assignments and interdependencies and tool-handling and material resources (Carroll, Neale et al. 2003) and makes the group's activities harder to understand than that of a single user's activity. This context is in the domain of user activities.

Group activity implies knowledge of how task components are identified, coordinated and carried out. Task components must be understood and pursued in the context of the overall purpose of a shared activity, the goals and requirements for completing it, and how individual tasks fit into the group's overall plan.

Table 7.2 Possible Activities Based on Location in the University Organisation

Room type	**Possible activities**
HoD office	Working on computer Talking on telephone *Having a guest* Staff meeting Student meeting
Administrator Office	Administration task Working on computer Talking on telephone *Having a guest* Staff meeting Student meeting
Staff/Lecturer Office	Working on computer Talking on telephone *Having a guest* Staff meeting Student meeting
Seminar	Meeting
Classroom	Teaching Student study group
Computer labs	Computing work Printing
Reading room/library	Finding book/journal/thesis Reading book/journal/thesis Take a note
Resources room	Printing Fax Stationery Mail box/pigeon hole Binding
IT support	Backup data Talking on telephone Burn CD/DVD Printing colour Server and workstation problem Network problem
Corridor	Walking Running
Student Study groups	Student Study group
Staff Meeting room	Meeting
Toilet	Toileting
Cafe	Morning Tea Afternoon Tea Lunch
Stair level 1	Pass through Walking/running
Stair level 2	Pass through Walking/running

When collaborative activity is carried out and the collaborators are in different time zones or cultures, it will not include face-to-face interaction and many interaction resources will be disrupted i.e., field of view is reduced, the possibility of gesture use is limited, facial expressions are eliminated/constrained, auditory cues are diminished, tools and artefacts cannot be as easily shared, deixis and spatial co-references are difficult to resolve (Tang 1991; Gutwin and Greenberg 1996). It is also difficult to repair miscommunication.

7.5 "Having a Guest" Using Mobile Access Point

How an Active Office deals with an "unregistered" user or unknown device and how a guest user, a user without enough knowledge of the local network, gains access to the Active Office is a unique situation in a social model of an Active Office. This will not only have implications for a context-aware application in providing service to the unregistered user, if it is allowed, but also to the development of the social policy/model of the environment as well, as a unique perturbation situation in an Active Office. The generic social scenario for this situation is when a staff member has a guest.

Generally when a guest user/colleague comes to the office, he is under the responsibility of the staff member, especially if the guest has limited access to office resources. How Active Office provides support for a guest user to office resources is a common problem. For that purpose, it must include the policy of the office, how confident the office administrator is of the network's security when it is accessed by an external user, such as a guest user, or other guest user categories (every guest user has his own characteristics and needs).

"Having a guest" is a unique user activity in an Active Office. The guest user could have limited access to the resources in an Active Office, which is unfamiliar domain for the user, and for this purpose a mobile access point for guest users is introduced in this section to provide access for the guest through a staff member's connection to the network. In an Active Office, a guest is a different user category from that of a visitor, such as visiting fellow, a guest may visit only for a couple of hours but a visitor may stay for several days and as a consequence of this situation, a visiting fellow may register in a local server, but a guest clearly may not.

The social policies designed for "having a guest" in an Active Office follows the requirement below:

- A user (staff member) who is employed in an Active Office is allowed to have a guest/colleague.
- A user guest is under the responsibility of the staff member.
- Active Office provides support for a guest user to gain access to limited general resources but , for security reason, a guest cannot directly using his workstation.
- Mobile computing equipment (Notebook or PDA) of a staff member can be used as a Mobile Access Point for his guest user in accessing Active Office resources.
- A Mobile Access Point can be used by the staff member and his guest user while they are on the move in a hot-spot wireless network area in an Intelligent Environment.
- At the same time, an Intelligent Environment application can locate a Mobile Access Point that turns to approximate guest user location.

In the implementation, a staff mobility device is set up as a mobile access point. This mobile access point is an access point for a guest user device to access the network. Staff members and their user guests can access the resources in the four buildings and the surroundings which are covered by the wireless network. While they are on the move in the Intelligent Environment, these resources can be accessed anytime, anywhere.

The proximate location of a guest user can be estimated by finding the proximate location of the mobile access point and turned to a proximate guest user location by using the symbolic or coordinate user location algorithm mentioned in Chapter 5.

The following is an example scenario of "having guests" in an Active Office.

> *John is a project leader in a joint project between industry, university and government. He has a smart personal assistant (SPA) in the form of a laptop with Linux Fedore Core 3 that uses wireless connection (WiFi and Bluetooth capable device) to the networks in his Active Office. When Adrian, Mick and Walter come to John's office and they bring their own SPA: Adrian brings his Phone PDA with Windows CE, Mick brings his PDA with Linux Familiar and Walter has a smart phone with Symbian operating system, all with Bluetooth capability. When they have an important meeting and need connections to the net, they should be able to create an ad hoc network through John's laptop using Bluetooth network even though they have different types of devices and operating system platforms. John's laptop acts as a mobile access point using Bluetooth network for his guests, but it remains as a client of the Active Office server using WiFi network.*
>
> *When Adrian has a connection to the Active Office network, the MAC address of his device is caught up by the resolution server to find his profile including his identity from his local server which is in a university server (outside John's office). The Active Office can then deliver a welcome message to Adrian and the meeting begins. The same thing happens to Mick from an industry server and Walter from a government server.*
>
> *The Active Office provides the mobile access point through John's laptop, calculating the guests' incoming and outgoing data which is stored as John's responsibility. The handling systems or Persistent-URL systems is used as resolution server to recognise the guest identity if his device is registered in his local resolution server. Moreover, the Active Office server can deliver information relating to John's location to his laptop which can lead to the location of their guests.*

Communication between small embedded devices and sensing devices are an integral service of the Active Office. The existing communication technologies use wired and wireless local area networks to form the communication network. The wired local area network uses fibre optic, RJ45 or USB networking and the wireless local area network uses WiFi, or wireless personal area networks (Bluetooth or IrDA). In an Active Office, the use of mobile phone network (GSM/GPRS) by the guest is allowed but not supported, since the access to mobile phone network is not under control of the Active Office infrastructure and the cost of connection also need to be considered. However, a guest has an option to connect to the network through 3G telecommunication service, for example, without any interference of Active Office systems.

In this implementation, several access points of Bluetooth and WiFi are used to cover the three buildings and their surroundings (see Figure 6.2 Chapter 6). The users are attached to the Active Office servers through the WiFi Access Points using Virtual

Private Network (secure mode), and the server provides authoritative dynamic IP address service by registering the user's MAC address devices for security and zero-configuration purposes. It is possible for an SPA user to connect to a WiFi server using wired or wireless LAN. When a user uses WiFi in an Active Office, his mobile device will be set up as a secure client to the WiFi's server to provide a connection for his guest, and his mobile device also needs to be set up as a Network Access Point (NAP) especially when his guest needs to connect using Bluetooth networking (Figure 7.3). When a guest brings his PDA with Bluetooth or USB capability, a scattered Bluetooth network or USB network can be built to access the Active Office resources.

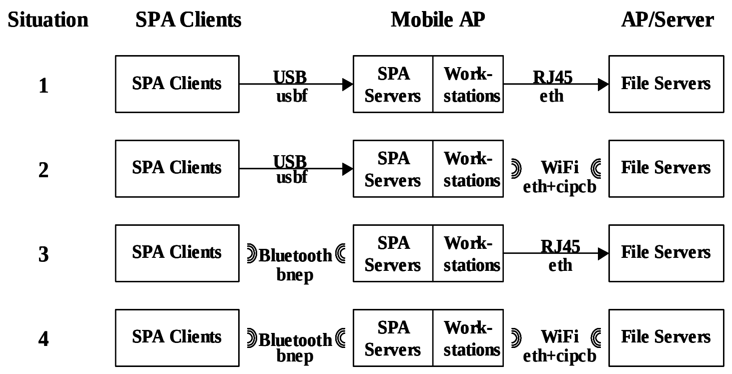


Figure 7.4 The Possible Connectivity of a Mobile Access Point to File Server.

In this experiment, two PDAs (WinCE and Linux Familiar) and a Smart Phone are used as a client and a laptop (Linux Fedora) to build USB networking or Bluetooth networking. While using Bluetooth Networking, the laptop was set up as a Bluetooth NAP and the PDAs and a Smart Phone were set up as a Bluetooth Personal Access Network User client (PANU).

From the study of the situation above, it is obvious that the user location for the SPA client depends on the location of the mobile access point. If the mobile access point moves then the SPA client will also move. Table 7.3 shows five possible situations (grey shading) which can occur when the SPA client comes to the proximate user location, while the rests are in fixed/precise user location. This happens when a mobile access point or an SPA client is using a wireless connection (Bluetooth/WiFi). The guest activities in accessing the resources while the guest user is on the move were monitored using network monitoring application.

To develop an application on the mobile access point to send user location data to clients, the Bluetooth, WiFi, USB and RJ45 connection can be considered as a regular connection in the Intelligent Environment. The four situations in which the connection to the Active Office can be formed are as follows (Figure 7.4):

1. SPA client uses USB Networking (usbf) to connect to a mobile access point and the mobile access point uses RJ45 (LAN/eth0) to connect to the File Server.
2. SPA client uses USB Networking (usbf) to connect to a mobile access point and the mobile access point uses WiFi (eth1+cipcb0) to connect to the File Server.
3. SPA client uses Bluetooth Networking (bnep) to connect to a mobile access point and the mobile access point uses RJ45 (LAN/eth0) to connect to the File Server.

4. SPA client uses Bluetooth Networking (bnep) to connect to a mobile access point and the mobile access point uses WiFi (eth1+cipcb0) to connect to the File Server.

Table 7.3 SPA Client Location Category.

Category	SPA Client	Mobile AP	AP/Server
1	Fixed/Precise Location	Fixed/Precise Location	Fixed/Precise Location
2	Mobile/Proximate Location	Mobile/Proximate Location	Fixed/Precise Location
3	Mobile/Proximate Location	Fixed/Precise Location	Fixed/Precise Location
4	Mobile/Proximate Location	Mobile/Proximate Location	Fixed/Precise Location

An Active Office provides full support to any resource movement (user mobility devices), in the sense of the availability of the independent resources. In the case of a mobile access point, it is possible for the staff member to change his connection from wired to wireless connection, but his guest continues to access the service transparently through his mobile access point, which is also his mobile device.

7.6 System Monitoring User Activity in an Active Office

Contexts have been defined as complex, rich objects that contain information relevant to the problem or domain being examined. This information is used to characterise the situation of a user entity and an environment entity, where the user entity is a person in the environment and an environment entity is an open or closed space that is embedded/equipped by smart sensors.

When a user has an activity in an Active Office, which is recognised by any sensors being in active use by a user or where any transaction occurs between a user and smart sensors in the environment, the user activity can be detected and monitored. The user activity information can be used to understand the user situation.

To monitor user activities, several important variables need to be considered, for example:

- user identification,
- user location,
- register of fixed devices/sensors,
- network availability (WLAN: Bluetooth, WiFi),
- service status of the room in the Active Office.

All objects, such as user identity, devices/sensors and network availability, have:

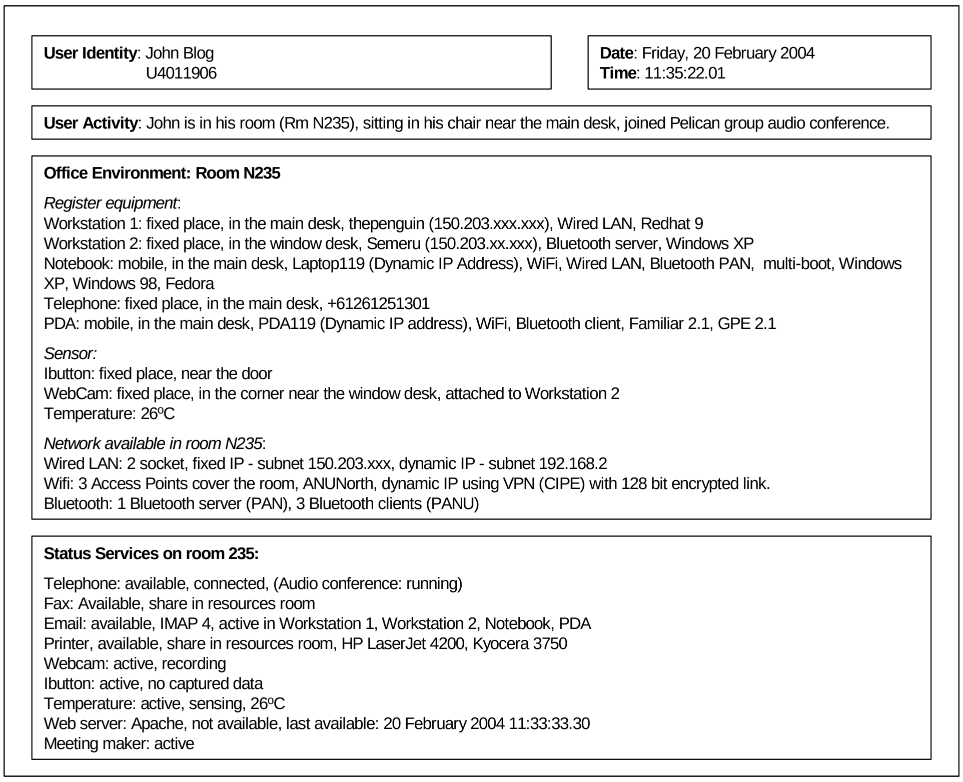
- object identification,
- an object name,
- other characteristics.

Once a relationship exists between user identification and objects such as user location or register devices, this relationship will be registered and stored in the Intelligent Environment repository as a transaction of a user model.

User location can be recognised by WiFi or Bluetooth. A proximity location sensor is used in the Active Office. In the case when the user has two devices with two connectivity capabilities, using WiFi and Bluetooth for instance, the Active Office environment will check both devices, then use the latest user location and store it in the Intelligent Environment repository as the current location.

Service status will be captured directly from the resources manager, which accesses the Intelligent Environment repository and the user model. The Intelligent Environment repository and the user model holds the information from all sensors/devices and the relationship between user identification and sensors/devices.

The user activity received the same treatment as the other device/sensor status, but is concerned only with user id and the transaction between user id and the objects i.e., sensors, devices, services.



User Identity: John Blog
U4011906

Date: Friday, 20 February 2004
Time: 11:35:22.01

User Activity: John is in his room (Rm N235), sitting in his chair near the main desk, joined Pelican group audio conference.

Office Environment: Room N235

Register equipment:
Workstation 1: fixed place, in the main desk, thepenguin (150.203.xxx.xxx), Wired LAN, Redhat 9
Workstation 2: fixed place, in the window desk, Semeru (150.203.xx.xxx), Bluetooth server, Windows XP
Notebook: mobile, in the main desk, Laptop119 (Dynamic IP Address), WiFi, Wired LAN, Bluetooth PAN, multi-boot, Windows XP, Windows 98, Fedora
Telephone: fixed place, in the main desk, +61261251301
PDA: mobile, in the main desk, PDA119 (Dynamic IP address), WiFi, Bluetooth client, Familiar 2.1, GPE 2.1

Sensor:
Ibutton: fixed place, near the door
WebCam: fixed place, in the corner near the window desk, attached to Workstation 2
Temperature: 26ºC

Network available in room N235:
Wired LAN: 2 socket, fixed IP - subnet 150.203.xxx, dynamic IP - subnet 192.168.2
Wifi: 3 Access Points cover the room, ANUNorth, dynamic IP using VPN (CIPE) with 128 bit encrypted link.
Bluetooth: 1 Bluetooth server (PAN), 3 Bluetooth clients (PANU)

Status Services on room 235:

Telephone: available, connected, (Audio conference: running)
Fax: Available, share in resources room
Email: available, IMAP 4, active in Workstation 1, Workstation 2, Notebook, PDA
Printer, available, share in resources room, HP LaserJet 4200, Kyocera 3750
Webcam: active, recording
Ibutton: active, no captured data
Temperature: active, sensing, 26ºC
Web server: Apache, not available, last available: 20 February 2004 11:33:33.30
Meeting maker: active

Figure 7.5 A Sample Snapshot of a User's Current Location and a User's Activity Recognition Window

The snapshot (Figure 7.5) is an example of monitoring of user activity. John Blog is in his room (room N235), sitting on his chair, near the main desk, and joins a teleconference with Pelican Group. John is monitored logging onto his computer (John is in room N234 and room N234 belongs to John), he sits on his chair (the iButton/RFID in the chair and keyboard activity is in active mode as John continues typing) and he registers onto the conference with Pelican Group (based on John's schedule, he is having a teleconference with the Pelican group and at the same time his phone status is in audio conference).

The context information above shows John's activity is a teleconference situation.

7.7 Summary

In the Context-Aware Computing/pervasive computing discipline, user activity is not well defined, this may be because the transition between activities is sometimes not very clear. An activity can be seen as a part of other activity, it can be in parallel or sequence mode or both together, and different views can be deduced from different activities.

User activity in the Intelligent Environment can be defined as any association between a user and smart sensors in the environment, or any sensors being in active use to access the resources. It is divided into 4 categories, i.e., an activity as association between a user and smart sensors in the environment, as a node in a work flow or job breakdown, as a physical movement mode or state, or as a mode of state of human intent.

Human activity is very complex and the computer environment is very limited in capturing user activity, but user activity is an essential ingredient for the determination of appropriate response/assistant behaviours to these activities in order to provide appropriate services without explicit commands. This leads to a situation-based approach in Context-Aware Computing.

In this chapter user activity processing based on DiCPA architecture is presented, followed by a case study of user activity in a University entity including a user's daily activity, the access zone and the case when a user has a guest. For this purpose, a Mobile Access Point is developed and analyses the gathering of the guest user location which leads to guest activity.

As the computing environment changes over time, monitoring user activity becomes progressively more difficult. In this study, a system monitoring user activity was proposed, based on the user activity processing model.

Chapter 8

PROVIDING INTELLIGENT RESPONSES IN A SMART ENVIRONMENT

How a smart environment has the capability to detect, respond and assist people with a variety of activities and to determine which self-decision actions to take based on the users' current context is a big challenge in the Pervasive Computing area. More over, as a consequence of users/objects continuously moving in the environment, handling a large volume of sensor data is also a problem. In responding to that problem, an Active Office was developed. It was equipped with fixed/precise sensors and proximate sensors (Bluetooth and WiFi) to locate and estimate user location. Further, a machine learning algorithm to estimate user location, which is described in Chapter 5, can produce a user location data to be stored in a spatio-temporal database technology. These approaches can be used to develop large-scale optimisation for the growth of sensor data. Both technologies, the machine learning and the spatio-temporal database approach, could be a basis/foundation technology used for the Active Office to respond based on the user's situation.

8.1 Introduction

In Context-Aware Computing, location information is the most important aspect in the environment, followed by activity information and response from the environment. Location information provides a context for user mobility, while user-activity information provides a context to identify the user's situation based on user location, with a response from the environment providing a context based on the user's situation, which is the action from the Intelligent Environment based on pattern matching from the sensors when sensing the presence of the user.

Space and time are the integral elements of the Intelligent Environment model, making dynamic spatio-temporal environments to enable transparent distributed computing to continually operate across changing circumstance in a seamless manner. There are two important problems in spatio-temporal environments, i.e., how to maintain consistent information about nearby objects while the user is on the move and how to process motion-specific queries from spatio-temporal sensors data in order to respond to rapidly changing aggregation of sensor data. Response to that problem sees the development of a scalable distributed context processing architecture which provides continuing operation across changing circumstances for users, the collection of nearby people and objects, accessible devices and the changing of those objects over time in the environment. This approach provides more understanding of user location, user activity and the kinds of intelligent responses coming from the environment.

The study began from understanding the user's presence in the environment, needing user identity and location identity, followed by user location which leads to user mobility, then user activity which leads to user situation, and response which leads to action in the environment.

User location can be identified by accessing available resources at static locations or by sensing the user’s mobile computing devices (PDA/handheld) while the user is on the move. Then, the study continues to include user mobility based on the user changing location from a current location to another location and storing user location sensor data as history data. History data can be reused and analysed to gain the pattern of user mobility. By understanding user mobility, there can be more understanding of user activity.

User Activity can be any association between a user and smart sensors in the environment or any sensors being in active use to access resources. Context information derived from user activity can be used to characterise the user situation.

By understanding the user’s situation, the environment has the capability to detect and respond to assist users by self-determining what action to take based on a users’ current context.

As an implementation model of Intelligent Environments architecture, an Active Office was developed consisting of several normal rooms with minimal intrusive detectors and sensors and not requiring user to wear badgets to make user-interaction with the environment easier.

The Active Office is equipped with two types of sensors, i.e., fixed/precise sensors and proximate sensors. Door sensor, chair sensor, phone sensor and keyboard activity sensor are used as fixed sensors with Bluetooth and wireless (WiFi) as proximate sensors to detect current state/context and to understand/recognise several contexts such as context location, context activity and context actions so that Intelligent Environment can provide service based on the user’s situations.

In identifying user identity and user location in an Intelligent Environment, the approach is to use the wireless connections in devices that users normally carry for other purposes, for example, a mobile phone or PDA. This device is not only handy because it is a small personal device but because it has a unique MAC address that can be used to establish user identity. The location of these devices, and hence the person with them, is determined by a mixture of precise, proximate and predicted location sensors. The data from these sensors is turned into a predictor to precisely locate the device, and thus the person. Once a user is located such services can be delivered based on the current situation from a resources manager.

8.2 Providing Responses in Context-Aware Computing

As a new approach of computing software engineering, *Context-Aware Computing* exploits rapid changes to accessing relevant information and the availability of communication and computing resources in the given environment of a mobile computer.

Context-Aware Computing could be considered as a basis or foundation for Ubiquitous Computing and Pervasive Computing, Ambient Intelligence and Nomadic Computing, as cognoscent of its user’s state and surroundings, and capable of determining what actions to take based on that context.

Context-Aware applications promise richer and easier interaction, but the current state of research in this field is still far from achieving that vision. The notion of context is still ill defined (Dey, Abowd et al. 2001).

Context-Aware Computing was implemented in a rich and rapidly changing computing environment, the Intelligent Environment. In this type of environment, most of the significant processes are done electronically by fixed/precise and proximate sensors while the user is on the move and the environment provides intelligent responses based on user context. The Intelligent Environment uses a scalable distribution context processing architecture (DiCPA architecture) to manage and respond to rapidly changing aggregation of sensor data (Kummerfeld, Quigley et al. 2003; Johnson, Carmichael et al. 2004; Mantoro and Johnson 2004).

DiCPA architecture provides task-relevant information to a user wherever the user location might be (Mantoro and Johnson 2004), while also having a deep understanding of activities in which the user is engaged. Demirdjian and Tollmar et al. worked on activity maps to detect a person's activity zone in a one-person office and a two-person office (Demirdjian, Tollmar et al. 2002). This establishes a correspondence between the zones to the "walking context", "working context", and "resting context" activities. Their approach has the capability of tracking groups of users but not of responding to user activity. The approach of enabling the Intelligent Environment to respond to user activity in this work can be divided into two categories, i.e. *guidance* and *assist/help* in an Intelligent Environment. Guidance means the Intelligent Environment provides step by step procedure/information for a certain purpose and the user will do the physical action. Assist/help means the Intelligent Environment will deliver a response to a user request with the physical response done by the Intelligent Environment. These intelligent responses to user activity are also inspired by Activity Theory (Engestrom 1987).

The trigger for the Intelligent Environment's responses can make the Intelligent Environment respond to a user's request or the pattern matching of the sensor data when the user is in an activity. This is an automatic response by the Intelligent Environment to the user. In either case, the response may be to assist (take automatic action) or to provide user guidance. The responses have four combinations of types of triggers, with some of the examples described in Section 8.6.

The taxonomy of context-aware system in responding to user activity is proposed as follows:

- **Sensing context** (the ability of the Intelligent Environment to detect and sense user activity from various devices and sensors).
- **Knowledge-based context** (the ability of the Intelligent Environment to develop relations between objects based on the rule-base for pattern-matching purposes and to understand the user activities, leading to the growth of the Intelligent Environment services).
- **History context** (the ability of the Intelligent Environment to develop patterns of user and environment behaviour based on the context's history).
- **Adaptation context** (the ability of the Intelligent Environment to learn and adjust the service structure based on user and environment behaviour patterns).
- **Response context** (the ability of the Intelligent Environment to interpret sensors' data to context appropriate services, the ability to determine what action to take based on user behaviour patterns in responding to current user activity).

8.2.1 Context as Predicate Relation

Context is about *predicate relation*: if there is no object, there is no context. To have a context, at least two objects have one relation or more. A single object rarely has a relation. Predicate relation may be thought of as a kind of relationship/function which applies to individual objects. When a group of static objects are approached by one or more moving object, it *spontaneously forms a context*. The behaviour of the context is more likely to have been formed in a group and ad hoc/spontaneous manner based on the patterns which match in the predicate relation.

Context could be defined as rich and rapidly changing predicate relations between objects (user and environment entity) that contain information relevant to the current local domain while an object (user entity) is on the move. This information is used to recognise the presence, location, mobility, activity and situation of the user entity as well as to respond and take action toward the environment entity.

User entity is a person in the environment, and the environment entity is an open (outdoor) or closed (indoor) space that is embedded/equipped with smart sensors. Most of the objects in the environment entity are located in a static location, while the objects on the user entity, such as a mobile phone or a PDA, move from one location to another followed by the users themselves. The objects on the users are devices that could be used as *user identity* in the environments. A user might have several devices that bring together a single user identity. A user might also have several identities from several devices. To have intelligent responses, a user must have as a minimum one device that leads to his identity, otherwise the Intelligent Environment has no capability to respond to his situation.

A context with quite rich relations can be categorised into two categories, i.e. user context and environment context. User context consists of attributes that have direct relation to the user, such as physiological state (e.g. body temperature and heart rate), emotional state, personal history, and so on. The environment context consists of attributes for servicing the user in the environment, such as physical user location, personalisation, navigation, and so on.

8.2.2 Presence

A user should be able to create a representative presence in the Intelligent Environment. All communication to a user is directed through this presence. A user may even create multiple presences. The presence brings a unique identification of the sensor, user and location. The user is in essence built into the network infrastructure. All communication and services destined for that presence are addressed to a unique identifier.

To have context, at least one user should be present in the environment. Without presence in the environment context will not exist. In other words, a user's presence is a minimal requirement to establishing context. Once a user is identified in the environment, the predicate relation can be formed. The context is proposed as having three types of key-predicate-relations awareness, i.e.:

- *Location Awareness* (the context of the environment to sense the object, deliver the object's location and form the relationship between the object and the surrounding object in the environment – such as "near me!" application).
- *Activity Awareness* (the context of the environment to sense the activity of the user in the user location and deduce the user's situation).

- *Response Awareness* (the context of the environment to detect the current user state/context and determine what actions to take based on user situation).

8.2.2.1 Location Awareness

Basically Location Awareness asks a simple question: Where am I? But in an Intelligent Environment that question reflects the user identity and the user location. Location Awareness brings two important concepts: Context Identity and Context Location. Both of the concepts have strong influences in the area of Location Object, User Mobility and Localisation.

Location Object deals with finding user location in the precise, estimated or predicted location. User Mobility deals with the change of a user's location from one place to another, if there is no change of service then there is no user mobility (Mantoro and Johnson 2003). Localisation deals with distributed data processing, i.e. the local data when the user is on the move.

In Intelligent Environments, the estimated location and predicted location are different concepts. *Estimated location* sees the use of proximate sensors to estimate user location. In the environment, a machine learning algorithm based on Self Organising Map is used (Mantoro and Johnson 2003) as well as a multivariate analysis approach to find the best match of user locations in the Active Office. Estimated location is used when fixed sensors cannot sense the user location in the Active Office. *Predicted location* sees the use of probabilistic method to predict current user location based on patterns of historical data of fixed sensors and proximate sensors data (Mantoro and Johnson 2003).

Context Location is a predicate relation between an object and its location. In this case, the object requests its position or location from the Intelligent Environment by using the "pull" method in the environment.

$$loc(object, location) \qquad \textbf{(1)}$$

The object itself can be a user, a chair, a phone, a door, a room or a building. For example:

loc(u1,room1)
loc(chair1, room1)
loc(room1, bldg108)
loc(bld10, ANU)

In the case where the object is a user, when he moves from one location to another, the predicate location could deduce its user's mobility.

$$\bigcup_{i=1}^{\infty} loc(object, location) = mobility \text{, for object as a person} \qquad \textbf{(2)}$$

A user could establish his location in the environment when he brings a mobile device, such as a Mobile Phone or a PDA with Bluetooth or WiFi enabled or both. The Bluetooth server as master Bluetooth will scan the user's Bluetooth enabled device as a slave/client. At the same time, when the user's WiFi enabled device is in a certain area, the signal strength and signal quality will be measured to get the estimated location.

loc(u1,room1):-
btloc(room1,u1,p800) .or.
wifiloc(PDA,u1,room1).

The ambiguity arising from the possible reporting of multiple locations for the user is resolved by the type of location sensors data and time. If the type of location sensors data is different, the priority of user location is precise location data, followed by proximate and predicted location data, if there are the same types of location data, then the latest data based on time will be used.

In the case of Bluetooth, the Bluetooth server is always "on-scanning" mode to scan the room until the client is found and then asks the resolve server to find the user identity (*carry*). The Bluetooth server then sends the user identity, location and the MAC address of the Bluetooth mobile phone.

btloc(room1,u1,p800):-
btscan(room1,p800) .and.
carry(p800,u1).

The environment tracks the mode of the mobile phone for user activity and responses by checking the status.

btstat (room1,u1,p800, "silent") :-
btloc (room1,u1,p800) .and.
mode (p800, "silent").

In the case of WiFi, the WiFi server is always in "on-listening" mode, when a mobile WiFi client reports signal strength and signal quality, it sends the estimation of user location, followed by resolution of the MAC address of the PDA which leads to user identity.

wifiloc(PDA,u1,room):-
w*ifiscan (PDA, room1),*
carry(PDA,u1).

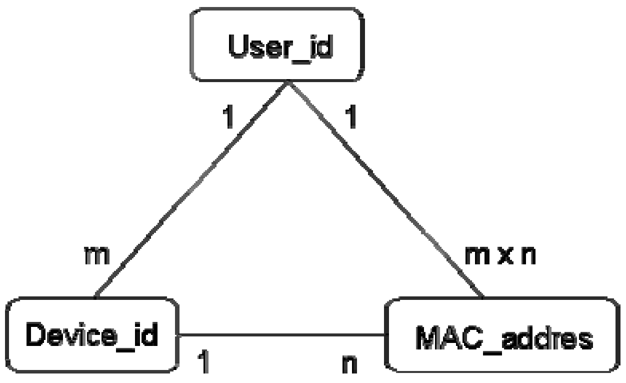


Figure 8.1 Triangle Resolutions: User Identification, Device Identification and MAC Address

A resolver database contains 3 basic components for user identity purposes; user identification, device identification and MAC address of the device. Figure 8.1 shows the triangle resolution used to resolve user identity when a user carries a device with a MAC address. It is possible for a user to have *m* (many) devices and a device to have *n* (many) MAC addresses implying that it is possible for a user to have $m \times n$ (many) MAC addresses. Section 7.3.3 describes the function of the resolver.

8.2.2.2 Activity Awareness

When a user knows his location, he will not need to ask "Where is my location?" The same thing happens when his colleague knows his location, thus the question will lead to "What are you doing?" However, in the Intelligent Environment, those simple questions can be interpreted as "What is your situation?" or "Can I find out your situation without intruding, interrupting or invading privacy?" The question is no longer simple and reflects the user's identity and the user's location.

Activity Awareness deals with how the environment understands user activity in a certain location. It has strongly influenced the concepts of Context Identity, and Context Location, which then lead to the concept of Context Activity.

Context Activity focuses on a predicate relation between the object's activities in a certain location. In the environment, object activities can be in the form of a user location mode, a person carrying a small device that he takes everywhere, or even a deduction made from the situation.

$$act(object, \{user \mid location \mid ded\}) \qquad \textbf{(3)}$$

User activity requires user location to recognise user activity hence user location is the subset of user activity. The Intelligent Environment senses all relevant objects in the examined location to recognise user activity.

$$\bigcup_{i=1}^{\infty} act(object, \{user \mid location \mid ded\}) = situation \qquad \textbf{(4)}$$

Activity awareness can be defined as any association between a user and smart sensors in the environment or any sensors being in active use to access resources. Context information from user activity can be used to characterise user situation. When an object has relationship to another object, context activity will be formed based on room scale. The Intelligent Environment has the capability to record all activities and, based on that collection of activities, a context situation can be deduced.

8.2.2.3 Response Awareness

Response Awareness focuses on how an Intelligent Environment responds to user situation. Examples of response questions include: Can I turn off all possible resources when there is no one in the room? Can I login or logout for you? Can I turn off the multimedia volume while you are on the phone? Or, can I set your mobile phone to silent mode while you are in the meeting? In an Intelligent Environment, all the responses are under a sequence in the clear control of the user. User control can be a direct or an indirect response. The first question above requests an indirect response and the last three questions ask for a direct response. A direct response needs direct approval from the user before it can be executed.

Response Awareness is different from Location or Activity Awareness, above, in the sense that it does not identify and inform an object's situation by requesting a response from the Intelligent Environment. Instead it uses 'pull' technology to deliver a response to the object/user.

Response Awareness reflects user/object identity, user location and user activity which lead to environment responses. In addition, it also brings and combines the concepts of Context Identity, Context Location and Context Activity to Context Response that allows the environment to deliver action for the user. The environment responses are carried out by any object in the environment other than user (*envi*).

Table 8.1 Summary of the Context-Aware Concept

Awareness	Core Concept	Context	Core Context	Key Word	Activity
- Presence Awareness	- Identity	- Context Identity	- Personalisation	- Who	- Who are you?
- Location Awareness	- Identity - Location	- Context Identity - Context Location	- Localisation - Location Object - Mobility Awareness	- Who - Where	- Where are you?
- Activity Awareness	- Identity - Location - Activity	- Context Activity	- Navigation - Situation Awareness	- Who - Where - What	- What are you doing? - What is your situation? - Can I find out your situation without intruding, interrupting or invading privacy?
- Response Awareness	- Identity - Location - Activity - Response	- Context Response	- Action Awareness	- Who - Where - What - Response	- How Intelligent Environment responds to user situation. - Can I login for you? - Can I turn off the multimedia volume while you are on the phone? - Can I set your Mobile Phone to silent mode?

Context response deals with a predicate relation of the pattern matching of the object activities in a certain location in responding to user situation. The response to the object in the environment can be deduced from such a situation.

$$resp(envi,user) \qquad \textbf{(5)}$$

When a certain location in the environment has both static and mobile objects, it can learn and adapt to deliver a response while keeping on monitoring activities, since the Intelligent Environment has the capability to record all the responses that are being delivered in which a context action from the environment to a user or a room scale situation can be deduced from a collection of responses.

$$\bigcup_{i=1}^{\infty} resp(envi, user) = action \qquad \textbf{(6)}$$

A summary of the concepts of Presence Awareness, Location Awareness, Activity Awareness and Response Awareness is shown in Table 8.1.

8.3 Sensor Management

To have a precise understanding of user presence, user location, user activity and response to user, the Intelligent Environment needs to understand the behaviour of every sensor, user identity structure, place and location structure. The strengths and weaknesses of every single object need to be understood. All the knowledge above has to be put in a context aware application, especially on a sensor structure of fixed sensors (chair sensors, phone sensors, door sensors and keyboard activity sensors) and proximate sensors (Bluetooth and WiFi sensor). The pervasive computing environment which deals with large volumes of sensor data needs to design fusion of the sensor data to have a sequence of clear logic of presence awareness, location awareness, activity awareness and response awareness.

8.4 Fusion Sensor Database Design

Pervasive computing by nature deals with large volumes of sensor data. As time goes and pervasive applications grow, the amount of data that is collected, stored and processed will steadily grow as well. This is likely to make the application drop speed in responding to user needs. The challenge of this problem is how to develop large-scale optimisation for the growth of sensor data which has a reasonable access time for a user. To optimise this, approaches are as follows:

1. Design a spatio-temporal database for the sensor's data.
2. Generalisation of the sensor data format of precise and proximate location sensor's data.
3. Filter the sensor data: record only when the state of the sensor is being changed.

8.4.1 A Spatio-Temporal Database for Various Fixed and Proximate Sensor's Data

In an Intelligent Environment, the sensor server application pushes the sensor data to the database when the state changes. Sensor databases are temporal in nature and rely on databases which record time referenced data (*temporal databases)*. A database is considered temporal if it is able to manage time-varying data and if it supports some time domain distinct from user-defined time. In temporal databases time can be captured along two distinct time lines: transaction time and valid time. The combination of both time databases is called a bitemporal database (Dyreson, Snodgrass et al. 1995; Jensen 2000). A spatio-temporal database is a special temporal database with extension of spatial databases. A spatio-temporal database has all the features of temporal databases and different key spatial attributes. As an extension of spatial databases, it also records moving objects (not static) as well as looking at any timestamp (Dyreson, Snodgrass et al. 1995; Abraham and Roddick. 1999; Parent, Esteban et al. 1999).

The spatio-temporal data model is a model where the space and time dimensions are the integral elements of the model. *Space* is the set of all spatial elements i.e., simple areas (rooms) and complex areas (building or city) contained in space type data. *Time* is the set of all temporal elements i.e., intervals, sets of disjoint intervals, sets of instants, heterogeneous sets of disjoint intervals and instants contained in time type data.

8.4.1.1 Mobile Objects Queries

Queries over mobile objects are almost similar to queries in traditional databases over static objects. The queries involve projection, selection, join, and aggregation operations. For the building block of complex SQL query, it uses selection queries, which can be classified as either a range (range predicate) or a nearest neighbour queries (*k*-Nearest Neighbour).

In range predicate, the objects that fall within a given spatio-temporal range are retrieved. In *k*-Nearest Neighbour, the objects that are relatively "closer" to the query point are selected. The notion of proximity differs between time and spatial dimensions. In the temporal sense, proximity means co-occurrence within a certain time period. Spatially, it refers to geometrical/geographical closeness based on sensor's data.

Thus, it is highly desirable to separate the selection predicate into spatial and temporal components, each of which can be specified in a different manner (Dyreson, Snodgrass et al. 1995; Parent, Esteban et al. 1999), either as a *k*-Nearest Neighbour or a Range predicate.

A Spatio-temporal Range Query is specified by both the spatial and the temporal ranges (Dyreson, Snodgrass et al. 1995). An example: *"retrieve all people (Bluetooth enabled mobile phone) in the corridor at level 2 of the building (spatial location range) where a director's meeting is taking place between 10-12am (temporal range)"*. A Temporal *k*-NN Query is specified with a spatial range and a nearest neighbour predicate on the temporal value (Dyreson, Snodgrass et al. 1995). An example: *"retrieve the first ten people (WiFi enable PDA) who were in the corridor between rooms E213 and E218 (spatial range) from the meeting room and ordered based on time differences between when the director's meeting occurred and the people crossed the spatial region (temporal k-NN predicate)"*.

8.4.1.2 Partition/Division Spatio-Temporal Database

The majority of database systems store information on real world events or the history of their objects. Such a real world event is then called the database subject. It usually happens that a certain database is partitioned/split into a few component units in order to reflect a natural order of events and the structure of its subject (Porkaew, Lazaridis et al. 2001). An example of the component unit is a class or an object in an object-oriented database system or a relation in a relational database model, as long as the design of the database follows at least the third normal form in the relational data model.

The database is in a third normal form if and only if the non-key attributes/fields are mutually independent and irreducibly dependent on the primary key. A non-key attribute is any attribute that does not participate in the primary key of the relation concern. Two or more attributes are mutually independent if none of them is functionally-dependent on any combination of the others. Such independence implies that each such attribute can be updated independently of all the others (Date 1995).

In the case of a distributed database system the division into units of component units is usually very clear. Data fragments stored in each site can be understood as particular component units and the subjects of these data fragments as certain parts of the real world. Processing the data related to such a separate part of the world or an environment can be carried out independently, provided that the actual autonomy of program application and data collection is realised in each site of the system (Porkaew, Lazaridis et al. 2001).

8.4.1.3 The Design of the Sensor Database

As mentioned earlier, in pervasive computing the amount of data that is collected, stored and processed from fixed and proximate sensors will continue to grow quickly, for example, it could reach 19203205 byte in 14 hours, for one sensor, which is a WiFi sensor in this case. This makes the speed of the application drop dramatically in responding to the user query. When the user requests access to the database, the speed of response access is a priority, especially in designing the temporal sensors database. The approaches are:

- to design the sensor database in third normal form and to develop a spatio-temporal database to speed up the user request.
- to record the occurrence only when the state of the user changes.

A simple example of designing the sensors' database is as follows:

Tables:
Person(*UserId*, Name, Phone, FaxNo, Mobile, Email, … , Address)
Location(*LocationId*, RoomNo, BuildingNo, DeptId)
Sensor(*SensorId*, SensorName, Type, Desc)
Device(*DeviceId*, DeviceName, Type, MacAd, Desc)
Database Transactions:
SensorDB(*SensorId*, *UserId, LocationId,* Time, State)
ResolvDB(*UserId, DeviceId, MacAd*)

In any design of the database for the sensor, the third normal form in the relational data model is the minimum requirement which should be followed.

Since the sensor database grows very quickly, in order to speed up the process of responding to a user query, spatio-temporal database transactions use partitions to split the database into generic databases for user requests. The approaches for partitioning the database are as follows:

- temporal database based on time,
- temporal database based on the sensor, and
- the combination of both approaches.

Developing spatio-temporal database transactions based on time can be achieved by creating several databases with the same structures, which create records based on time. This approach has a redundancy tuples occurrence in the database transaction, however the size is not that large, since there is a filter based on time. It still has a reasonable time to search or match the data item in responding to a user request or to the web server to pull the data to show in the graphs. Done this way, the access time required to pull the sensor data is faster.

Consider the sensor database within office hours (9am-5pm), within a day, a week, a month or a year. For example:
SensorDB0917(*SensorId*, *UserId, LocationId,* Time, State)
SensorDBaDay(*SensorId*, *UserId, LocationId,* Time, State)
SensorDBaWeek(*SensorId*, *UserId, LocationId,* Time, State)

SensorDBaMonth(*SensorId*, *UserId, LocationId,* Time, State)
SensorDBaYear(*SensorId*, *UserId, LocationId,* Time, State)
SensorDBall(*SensorId*, *UserId, LocationId,* Time, State)

When the fixed and proximate sensor servers add a record, they record to several relevant databases.

A spatio-temporal database transaction based on a sensor can be developed by creating a number of databases with the same structure but with records based on sensors. Each sensor has its own database. When the fixed or proximate sensor server adds a record, it writes to a single temporal database sensor.

For example:

SensorDBEA003(*SensorId*, *UserId, LocationId,* Time, State)
SensorDBEA004(*SensorId*, *UserId, LocationId,* Time, State)
…
SensorDBEA099(*SensorId*, *UserId, LocationId,* Time, State)
SensorDBEA100(*SensorId*, *UserId, LocationId,* Time, State)

Both approaches above can be combined by developing temporal databases based on time and sensors. For instance, first, a relational data model using spatio-temporal databases based on time is designed, and then it is decided which sensor is used frequently, which is then split/partitioned into other temporal databases for that particular sensor only.

For example:

SensorDBaDay(*SensorId*, *UserId, LocationId,* Time, State)

This is a temporal database sensor for a day (SensorDBaDay) and if there are 4 out of 100 sensors data in the database that are used very actively, then it can be split into four other sensor data databases.

For example:

SensorDBaDayEA003(*SensorId*, *UserId, LocationId,* Time, State)
SensorDBaDayEA004(*SensorId*, *UserId, LocationId,* Time, State)
SensorDBaDayEA099(*SensorId*, *UserId, LocationId,* Time, State)
SensorDBaDayEA100(*SensorId*, *UserId, LocationId,* Time, State)

The rest will remain in the database SensorDBaDay, which contains the total sensor data.

8.4.2 Generalisation of the Sensor Data Format

An Active Office is equipped with two types of sensors, i.e. fixed sensors and proximate sensors. Fixed sensors use door sensor, chair sensor, phone sensor, RFID sensor, keyboard activity sensor, and mouse activity sensor, the proximate sensors use Bluetooth and wireless (WiFi).

The processing for both fixed and proximate sensors is shown at Figure 8.2. The fixed sensor server records all object-IDs from fixed sensors. The proximate sensor server records all object-ids from proximate sensor devices such as a PDA, a Mobile Phone or a Smart Phone that have Bluetooth or WiFi enabled. The location server delivers a symbolic user location based on the data from the WiFi sensor server and records every occurrence to an aggregation server. The resolution server converts the Bluetooth MAC address to user identity, gets the symbolic location from the Bluetooth proximate server and bundles it with the relevant object-ID. Once the object is formed, it is then recorded in the Aggregate Sensor Server.

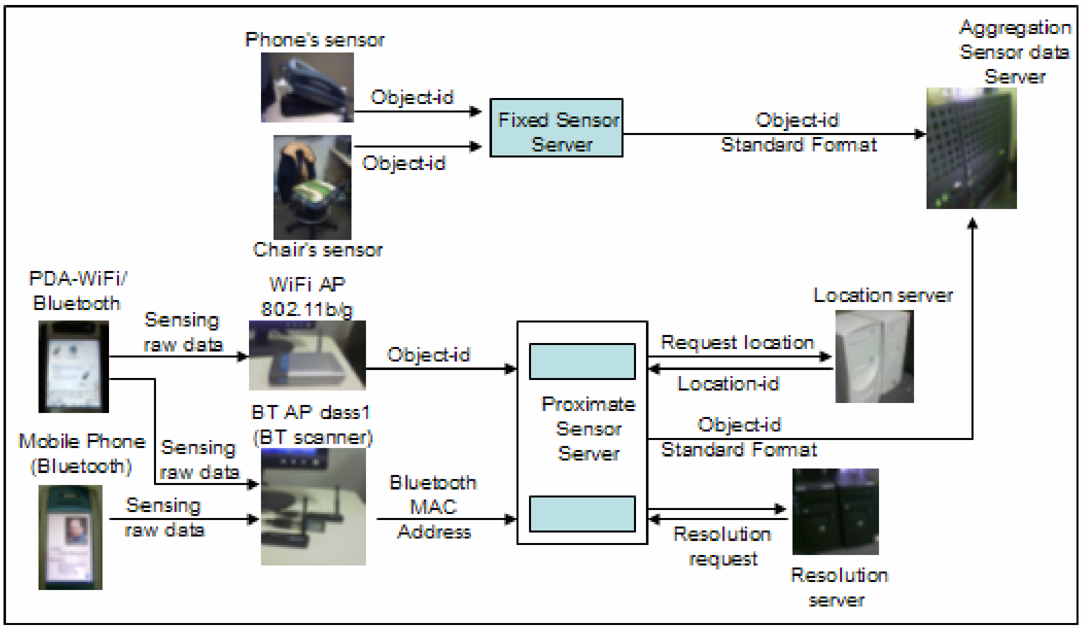


Figure 8.2 Smart Sensors Processing From Fixed and Proximate Sensors Server

A WiFi access point cannot be used to sense the WiFi client, but the client itself pushes WiFi object data to the proximate sensor server. The object contains information on signal strength, signal quality and MAC address as the object-id. A Bluetooth access point has different behaviour, it senses (scans) the area and pushes all Bluetooth objects found in the area to the proximate sensor server.

Every sensor object has its own sensor identity and user identity. For example:

Sensor ID	*User ID*	*Descriptions*	*Sensor type*
EA003	U4011906	Sensor embedded in Joe's Chair	Fixed/Precise
EA004	U0000000	Door sensor belongs to everybody	Fixed/Precise
EA100	U4011906	Joe's PDA	Proximate
EA099	U4011906	Joe's Mobile phone	Proximate

For every occurrence of sensing data, the standard input format is below:

standard format = { user-id | equipment-id, sensor id, location-id, time, sensor state }.

The user-ID data item is the user identity in a certain location, which can also be replaced for equipment identity. In this chapter equipment-ID is not used yet. Time follows a Unix format standard time and the state values are either 0 or 1. When the state is changed, occurrences are recorded in the database.

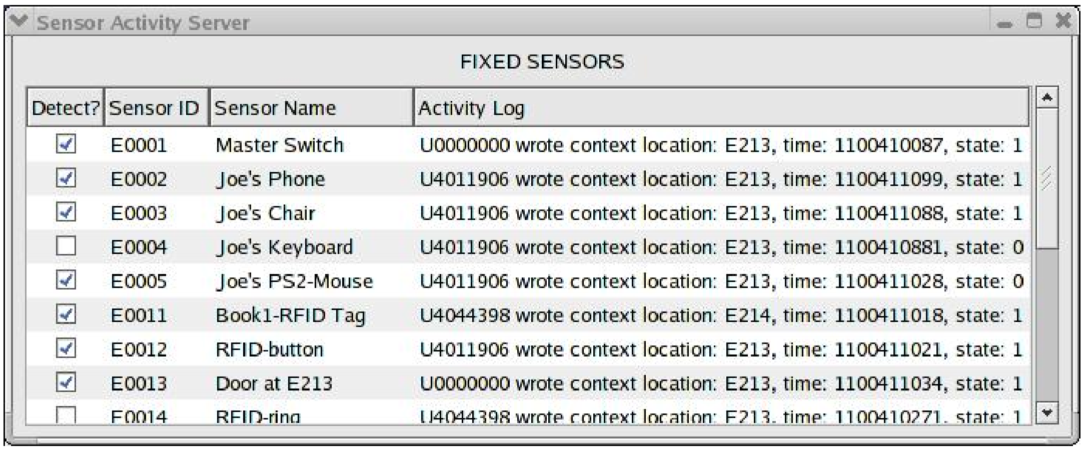
The fixed sensor server and proximate sensor server have different approaches to the recording of occurrences. Table 8.2 shows examples of sensor data and includes their interpretation.

Table 8.2 Sensor Data and the Interpretations

Sensor ID	User ID	Location ID	Time	State	Interpretation
EA013	U0000000	E231	1090974717	1	The door in room E231 is open and has been since 6:23:31 25/08/04, that was 5 minutes ago.
EA003	U4011906	E231	1090974717	1	Somebody is sitting in Joe's Chair in room E231 and has been since 6:23:31 25/08/04, that was 5 minutes ago
EA99	U4011906	E231	1090974717	1	Joe's Bluetooth enabled Mobile phone is in his room (E231) and has been since 6:23:31 25/08/04, that was 5 minutes ago
EA099	U4011906	E231	1090974717	1	Joe's WiFi enabled PDA is in his room (E231) and has been since 6:23:31 25/08/04, that was 5 minutes ago

The sensor server contains information such as: whether the sensors actively detect or not, the sensor-id, the sensor name, and the activity log. The fixed sensor server with the raw sensor data is shown in Figure 8.3. The process of recording the fixed sensor data can be illustrated as follows:

EA013 is a sensor-ID for the fixed sensor for a door of E231 room. Room E231 is occupied by 3 persons, so the EA013 sensor belongs to everybody (U0000000). EA013 is a fixed door sensor therefore the sensing data is pushed to the sensor database by the fixed sensor server. The default door sensor state is 0, it means that the door is closed. If someone opens the door, the state changes into 1, and it will add a record such as (EA013, U0000000, E231, 1090974717,1) to the sensor database through the wireline network.

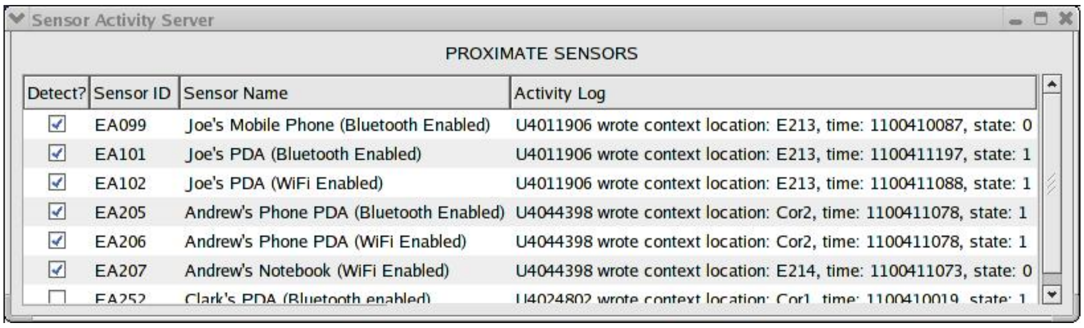


Detect?	Sensor ID	Sensor Name	Activity Log
✓	E0001	Master Switch	U0000000 wrote context location: E213, time: 1100410087, state: 1
✓	E0002	Joe's Phone	U4011906 wrote context location: E213, time: 1100411099, state: 1
✓	E0003	Joe's Chair	U4011906 wrote context location: E213, time: 1100411088, state: 1
☐	E0004	Joe's Keyboard	U4011906 wrote context location: E213, time: 1100410881, state: 0
✓	E0005	Joe's PS2-Mouse	U4011906 wrote context location: E213, time: 1100411028, state: 0
✓	E0011	Book1-RFID Tag	U4044398 wrote context location: E214, time: 1100411018, state: 1
✓	E0012	RFID-button	U4011906 wrote context location: E213, time: 1100411021, state: 1
✓	E0013	Door at E213	U0000000 wrote context location: E213, time: 1100411034, state: 1
☐	E0014	RFID-ring	U4044398 wrote context location: E213, time: 1100410271, state: 1

Figure 8.3 Fixed Sensor Server.

The proximate sensor servers, such as Bluetooth and WiFi, have different approaches to the sensing of data. Figure 8.4 shows the proximate sensor server data; there is a Bluetooth sensor server and WiFi sensor server. The WiFi sensor server is in listening mode, but clients such as Notebook or PDA with WiFi network enabled, will sense the access point and push the sensing data to the sensor database.

> For example, *EA101 is a sensor-ID for U4011906 user's PDA (PDA IPAQ 3970 with WiFi network card), when the user comes into room E231, the PDA will request his location to location server based on signal strengths and signal qualities of all WiFi access points in his location, change the state and pick up the current time, then send a record to be added into the sensor database, such as (EA101, U4011906, E231, 1090974717, 1) through the WiFi Wireless network.*



PROXIMATE SENSORS

Detect?	Sensor ID	Sensor Name	Activity Log
✓	EA099	Joe's Mobile Phone (Bluetooth Enabled)	U4011906 wrote context location: E213, time: 1100410087, state: 0
✓	EA101	Joe's PDA (Bluetooth Enabled)	U4011906 wrote context location: E213, time: 1100411197, state: 1
✓	EA102	Joe's PDA (WiFi Enabled)	U4011906 wrote context location: E213, time: 1100411088, state: 1
✓	EA205	Andrew's Phone PDA (Bluetooth Enabled)	U4044398 wrote context location: Cor2, time: 1100411078, state: 1
✓	EA206	Andrew's Phone PDA (WiFi Enabled)	U4044398 wrote context location: Cor2, time: 1100411078, state: 1
✓	EA207	Andrew's Notebook (WiFi Enabled)	U4044398 wrote context location: E214, time: 1100411073, state: 0
☐	EA252	Clark's PDA (Bluetooth enabled)	U4024802 wrote context location: Cor1, time: 1100410019, state: 1

Figure 8.4 Proximate Sensor Server.

The Bluetooth sensor server has a different approach from that of the WiFi server, the master Bluetooth, as a Bluetooth access point similar to the chair sensor or phone sensor, is fixed in certain locations. The Bluetooth access point senses location continuously, with sensing duration between 8-14 seconds. When the master Bluetooth captures the slave/client, it will request the resolution server for the identity of this object and push the sensing data to the sensor database.

> For example*, when a user comes into room E231 with a Bluetooth enabled Mobile Phone, the Bluetooth access point server, which senses continuously, will capture the approach of a Bluetooth slave. The Bluetooth access point then requests that the resolution server, based on slave Bluetooth MAC address, find the identity of the object such as sensor id and user id. Then the Bluetooth access point server changes the state and picks up the current time, and sends a record to add to the sensor database, such as (EA099, U4011906, E231, 1090974717, 1) through, the wireline network.*

8.5 Response to User Activity

In pervasive systems, location information is a very important aspect of providing a context for user mobility, e.g. finding the nearest resources, navigation, locating objects and people. User mobility has been studied in an Active Office (Mantoro 2003; Mantoro and Johnson 2003). This study began with understanding user location. User location can be identified in several ways, such as tracing user activity when accessing available resources at static locations or by sensing the user's personal mobile computing devices (PDA/handheld). Next, the process continues to user mobility, based on a user's changing location from the current location to another, with that information being stored in the history database. The history database is then analysed to obtain the pattern of user mobility. The study concluded by stating that by understanding user mobility, user activities in an Active Office can be better understood.

User activity can be defined as the association between a user and smart sensors in the environment, or any sensors which are in active use to access the resources. The context information from user activity can be used to characterise the user situation.

To monitor user activities, several important variables are needed, such as user identification, user location, registered fixed devices/sensors, network availability (WLAN: Bluetooth, WiFi), and service status of the room in the Active Office. The user identity, devices/sensors and network availability as objects would have object identification, an object name and any other characteristics. Once a relationship exists between user identification and other objects, such as user location or registered devices, this object will be registered and stored in the Intelligent Environment repository as a transaction in a user model.

User location can be recognised by WiFi or Bluetooth. A proximity location sensor is used in the Active Office. In case the user has two devices with dual connectivity capability, using WiFi and Bluetooth for instance, the Active Office environment will check both devices, then use the latest user location and store it in the Intelligent Environment repository as the current location.

The service status will be captured directly from the resources manager which accesses the Intelligent Environment repository and the user model. The Intelligent Environment repository and the user model hold the information from every sensor/device as well as the relationship between user identity and latest sensor sensors/devices data.

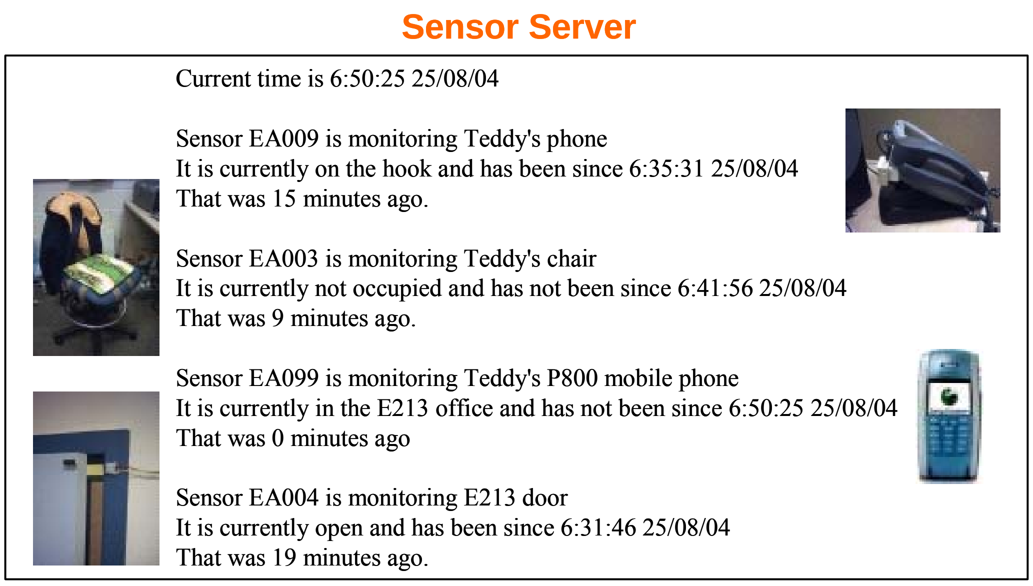
User activity uses the same procedure as user location but it needs user identity, the sensor/device, location id, date time and status of services.

8.6 Modelling Social Environments: Responding to User Situations

Context is central to social interaction. It is not just simply interaction between an individual user and a computer system. Its broad interaction merges with the social, home/work culture and organisation factors that affect the interaction. The user makes decisions on the actions and the Intelligent Environment interprets the system's response. When the context is central to social interaction, the sensor is the interface for the interactions of the user and the environment.

Any responses from the environment that have to deliver to users need user identity. The problem is that not all sensors can be represented as user identity. The sensors/devices that can be used as user identity in the Active Office have two requirements:

- They are used as personal and private sensors/devices.
- They have connectivity to the network, such as Bluetooth or WiFi enabled sensors/devices.


Sensor Server

Current time is 6:50:25 25/08/04

Sensor EA009 is monitoring Teddy's phone
It is currently on the hook and has been since 6:35:31 25/08/04
That was 15 minutes ago.

Sensor EA003 is monitoring Teddy's chair
It is currently not occupied and has not been since 6:41:56 25/08/04
That was 9 minutes ago.

Sensor EA099 is monitoring Teddy's P800 mobile phone
It is currently in the E213 office and has not been since 6:50:25 25/08/04
That was 0 minutes ago

Sensor EA004 is monitoring E213 door
It is currently open and has been since 6:31:46 25/08/04
That was 19 minutes ago.

(http://vonneumann.anu.edu.au/uaie/uaie.php)

Figure 8.5 Sensor Server for the Active Office

Small devices, such as a Bluetooth or WiFi enabled PDA or Mobile Phone, can be used as user identity in the Intelligent Environment.

Other sensors such as chair sensors, door sensors or phone sensors cannot be used as user identity. A chair with an embedded sensor on it cannot be used as user identity, because it is possible for any person to sit on it. The same possibility of multiple use applies to phone or door sensors.

An Active Office uses several sensors (Figure 8.5) as infrastructure in developing an Active Office model to enable it to respond to the user's situations. The Active Office has to recognise the current state/context of a user's situation and respond, giving guidance or assistance/help based on daily user activity.

In the following section, the social environment model which enables and responds to a user's situation for the Active Office is described by the following three scenarios:

- the meeting scenario,
- the automatic login/logout, and
- how the environment responds when a user has a phone call.

8.6.1 When There is a Meeting

In a regular office, when there is an important meeting, its participants do not want to be interrupted either by something from inside or outside of the room. In the meeting room scenario, the room should have the capability to understand the activity and the situation in the meeting room and take action when there is an interruption/disturbance from outside, but let the user himself control the situation. For example, when there is such a meeting (recognised by the closed door, most participants are sitting in the room with most mobile phones on silent mode), the Active Office has the capability to anticipate situations that may interrupt/disturb the meeting, such as the ringing of mobile phones or knocking on the door from the outside.

In the meeting room scenario, assume that the meeting room is equipped with:

- *6 chairs with sensor chairs embedded,*
- *1 door with door sensor embedded,*
- *Bluetooth sensor scans in the meeting room area, and*
- *WiFi sensor covers the building including the meeting room*

The environment first finds the 'presence' of an object in the meeting room by using the predicate relation 'status' to check if there is a new changing state:

- *Stat(door,"close")*
- *Stat(chair1,"occupied")*
- *Stat(chair2,"not occupied"*
- *Stat(chair3,"not occupied")*
- *Stat(chair4,"not occupied")*
- *Stat(chair5,"occupied")*

After that, the proximate sensor server finds the latest state of user location in the meeting room (finding the presence of users):

- *BTscan(P800,u2,room1)*
- *BTscan(N6600,u3,room1)*
- *BTscan(P900,u4,room1)*
- *WiFiscan(PDA1,u1,room1)*
- *WiFiscan(PDA2,u2,room1)*

When the environment has found 3 of the Bluetooth enabled devices, for example, then the Active Office's Bluetooth scanner checks the 'mode' of the mobile Bluetooth clients in the meeting room:

- *Mode(P800,"silent")*
- *Mode(N6600,"silent")*
- *Mode(P900,"active")*

Subsequently the Active Office delivers the information based on the sensor data, as below:

The users in the meeting room *(Context Location)* are:

- *U1 is (PDA) in the meeting room*
- *U2 is (PDA) in the meeting room*
- *U2 is (Mobile Phone) in the meeting room*
- *U3 is (Mobile Phone) in the meeting room*
- *U4 is (Mobile Phone) in the meeting room*

The activities *(Context Activity)* are:

- *Chair1 is occupied*
- *Chair2 is not occupied*
- *Stat(chair3,"not occupied")*
- *Stat(chair4,"not occupied")*
- *Stat(chair5,"occupied")*
- *Door is closed*
- *User1 (PDA) is in the meeting room*
- *User2 (PDA) is in the meeting room*
- *User2 (Mobile Phone) is in silent mode*
- *User3 (Mobile Phone) is in silent mode*

- *User4 (Mobile Phone) is in active mode*

The samples of generic rules to capture the context activity for a meeting scenario are as below:

```
Meeting_in_progress(user, dev, chair, door,room):-
    Less_than(User_not_sitting (use, dev, room),
    Chair_occupied(Stat(chair,"occupied"),2),
    Stat(door,"closed"),
    Check_BT_silent(Mode(user)),
    Broadcast(Write("Meeting in progress and cannot be
    disturbed."),department).
User_not_sitting(user,dev,room):-
    (Number_of_user(BTscan(dev,user, room), WiFiscan(dev,user,room)) -
    Chair_occupied(Stat(chair,"occupied")),
BT_not_silent(user, cmd_to_silent):-
    Check_BT_not_silent(mode(user), Send_cmd(cmd_to_silent, yes(_))).
```

The rules above can be described as *if* the number of not-sitting-user *less than* sitting-user *and* is *less than* two users, *and* the door is closed *and* all the mobile phones with bluetooth capability are in silent mode *then* the situation may be deduced as "*Meeting in progress and cannot be disturbed"*.

The situations (*Context Situation*) which are deduced from the rules of the context activities above are:

- *4 people in the meeting room*
  - *2 people sit on the chairs*
  - *2 person does not sit on the chair*
- *Door is closed*
- *3 Mobile Phones are in the meeting room*
  - *2 Mobile in silent mode*
  - *1 Mobile in active mode*

Based on the situations above the Intelligent Environment concludes that "*A meeting is in progress and cannot be disturbed*" and broadcasts the message to the relevant office area/people. Broadcasting the message in the department network is also an example of a response by the Active Office. Another possible response arising from the example above is where the context activity deduces that there are 4 people in the meeting room, finds that 1 of 3 Mobile phones is not in the silent mode, the Intelligent Environment then asks the user whether or not it is allowed to change the mobile phone to silent mode.

8.6.2 The Automatic Login\Logout in an Active Office

In a generic office it is regular behaviour for a user to login his computer or unlock the computer monitor to start work in the morning, and to lock the computer monitor for morning tea, lunch, or afternoon tea. Sometimes he may need to logout from his computer at other times in addition to at the end of his day in the office.

To enable an Active Office to respond to the user situations above, an embedded sensor in his chair and a Bluetooth scanner are used. When the user comes into a room in an Active Office and he brings his Bluetooth enabled PDA or Mobile Phone in discoverable mode, the Bluetooth Scanner, as the Bluetooth proximity sensor location, will scan and find that a user with his identity is in the room. Then, when he sits on the chair at his computer desktop, the chair's sensor informs the Active Office that someone is sitting on the chair. As the user's Mobile Phone is a trust device in the Active Office, it will respond by automatically presenting the last state of the user environment as the last time he worked on his computer desktop. For example:

Situation:

Bluetooth scanner found Joe's Phone in Room E213, and found someone sitting on Joe's chair.

Response:

Present the last screen situation when Joe logged out last.

More detail can be described as follows:

Current state:

Sensor Data:

("U4011906", "E213","EA099",1090973456,1) from Bluetooth scanner
("U4011906", "E213","EA003",1090990544,1) from Joe's chair sensor

Interpretation:

Joe and his Mobile phone came into the E213 room and this was followed by *someone* sitting in Joe's chair. The mobile phone sensor has Joe's identity but the chair sensor has no user identity containment.

The rules for sensor data pattern matching which activates the response agent:

```
Automatic_login(_,sessor_id):-
    Stat(sensor_id, "EA099"),
    Stat(SensorId, "EA003"),
    Less_than (Time("EA099"), Time("EA003")),
    Sensor_state("EA009", 1),
    Sensor_state ("State(EA003",1).
```

The response from the Active Office:

- Check if the desktop computer state is locked, and then unlock the computer as Joe does.
- If the desktop is not locked but in a ready state then login as Joe does and present the last screen situation when he logged out last.

The GUIs in current operating systems, such as Windows, Unix, Linux and Mac, have timeout processes. However, the operating systems cannot automatically respond to the user situation, whereas the timeout process responds to any user based on time only, without considering the identity of the user. The automatic login/logout in Active Office is the response from the environment to the user based on the user identity (Bluetooth scanner) and the user situation (chair sensor).

When a user has to leave his room with his mobile and his chair is not occupied for 10 minutes then all the work is saved, the desktop is locked, after an hour the computer will logout. Since there is only one chair in front of the desktop computer, it will allow a user to login, however any user with his trusted personal device and who also sits will have the same response: an automatic login and restoration of that user's previous state.

8.6.3 Response When a User has a Phone Call

A phone call for a user is usual in the generic office. People usually put a phone on the main table, close to the desktop computer. Today, the computer has a set of multimedia software that plays various format multimedia entertainments such as music, video clips or movies as a standard computer feature. When a user utilises multimedia entertainment and the phone rings, if the volume is not turned down or the multimedia player switched off, his conversation may be disturbed by inappropriate background sounds.

In an Active Office, the environment can be enabled to respond to the user situation above by reducing, pausing or even turning off the application that produces the sound. *ir*Media Player is a prototype which developed for this purpose. It communicates with an embedded sensor in the phone and a Bluetooth scanner. When a user receives a phone call, the state of the phone is on the hook to start with and since for every response an Active Office needs a user identity to which the user's response will be delivered, a user with his Bluetooth enabled PDA or Mobile Phone in discoverable mode should be in the room. The Bluetooth scanner as a Bluetooth proximity sensor location will scan the area continuously to understand that a user is present within the room, and therefore captures the user's identity.

When a phone rings and the user picks up the phone, the phone status is "off the hook". Simultaneous with the user taking the phone off the hook, the fixed phone sensor server will send an instruction to the desktop computer to appropriately reduce the volume of the multimedia sound (Figure 8.6a). Moreover, the fixed Bluetooth sensor server scans the availability of a user's PDA or Mobile Phone in the area. If the multimedia service is active, and produces surround sound, the Bluetooth sensor server automatically sends an instruction to reduce/pause the volume. For example:

Situation:

Joe's Phone sensor just changed to the status "off the hook".

The Bluetooth scanner scans Joe's Mobile Phone with discoverable Bluetooth mode in Room E213.

Response:

Reduce or turn off the volume in the desktop computer and also on Joe's Mobile Phone.

More detail can be described as below:

Current state:

Sensors Data:

("U4011906", "E213","EA099",1090973456,1) from Bluetooth scanner

("U4011906", "E213","EA009",1090994856,1) from Joe's phone sensor

Interpretation:

Joe and his personal mobile phone came into room E213 and *someone* took Joe's phone off the hook.

The mobile phone sensor has Joe's identity, but the phone sensor has no user identity containment.

The sensor data pattern matching which activates the response agent:

Response_to_phone_call(_,sensor_id):-

Stat(sensor_id,"EA099"),
Stat(sensor_id, "EA009"),
Sensor_state("EA099",1),
Sensor_state("EA009",1).

The response from the Active Office:

- Reduce or turn off the volume in the desktop computer.
- The Bluetooth scanner scans for Joe's Mobile Phone and searches for the multimedia volume and reduces or turns it off if it emits sound.

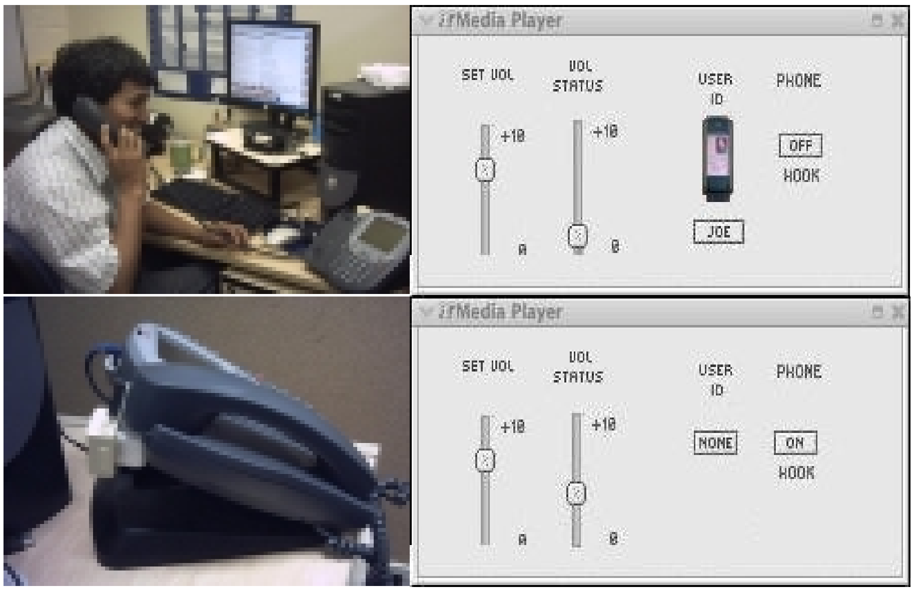


Figure 8.6 *ir*Media Player Monitoring Status

a) the phone is "off hook" and the multimedia player keeps the volume and volume as zero and,
b) the multimedia player is on and phone is "on hook".

When the user finishes and puts the phone handset back, the phone's sensor server will change phone status to 'on hook'. The phone's sensor server will then send an instruction to restore the volume of the desktop computer and Joe's Mobile Phone to their previous levels (Figure 8.6b).

8.7 Monitoring of the Sensor's Activity

To monitor and evaluate the sensor's activity in an Active Office, a graph to represent current sensors' activities was developed. The graph was based on current data for a day and is presented on an hourly basis (Figure 8.7). The sensor's data were recorded each time the sensors changed the status. Another spatio-temporal database was also developed for this purpose by recording two records for every single status change occurring in the sensor database. (At the time of writing, the current sensors graph can be seen at http://vonneumann.anu.edu.au/uaie/sensorgraph.php).

The recording of the database for development of the graph was as follows:

- The initial state for the embedded sensor on the chair was:
  (“U4044398”, “E213”,”EA014”,1090973686,0)
  It then changes the state because someone was sitting on the chair:
  (“U4044398”, “E213”,”EA014”,1090974717,1)
  after 10 minutes he got off his chair:
  (“U4044398”, “E213”,”EA014”,1090975317,0)

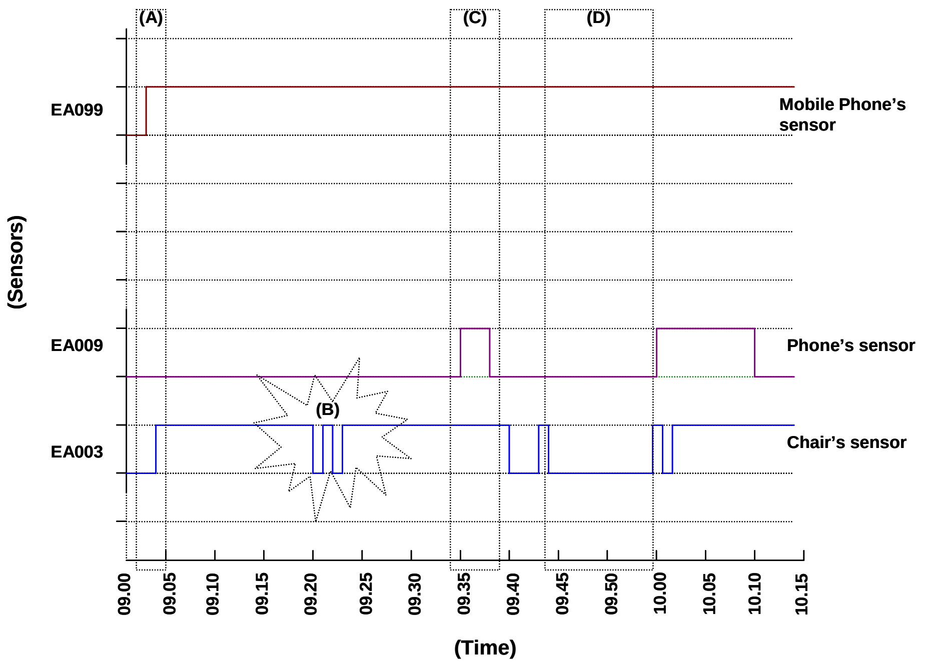


Figure 8.7 Monitoring the Fixed and Proximate Sensors’ Activity Graph.

- For the purposes of the graph, it recorded twice as follows:
  (“U4044398”, “E213”,”EA014”,1090973686,0) initial state
  (“U4044398”, “E213”,”EA014”,1090974717,0) time changes but state is the same
  (“U4044398”, “E213”,”EA014”,1090974717,1) state changes but time is the same
  (“U4044398”, “E213”,”EA014”,1090975317,1) time changes but state is the same
  (“U4044398”, “E213”,”EA014”,1090975317,0) state changes but time is the same

8.8 Summary

This chapter focused on designing a framework for an Intelligent Environment to respond intelligently to user activities in a context-awareness environment. First, a view on Context-Aware Computing is presented, followed by fusion sensor database design and the modelling of the social environment to respond to user activities.

In the concept of the Context-Aware Computing to form user context, the environment must recognise the physical *presence* of a user and in this chapter three key predicate relations are proposed as below:

- *Location Awareness* which leads to *User Mobility.*
- *Activity Awareness* which leads to *User Situation.*
- *Response Awareness* which leads to *Action of User Situation* from the environment.

This chapter contributes to Context-Aware Computing by:

a. describing the concept of Context-Aware Computing as a new software engineering approach which could deliver responses to the user.
b. proposing a solution to the problem of handling a large number and variety of fixed and proximate sensor data by using a spatio-temporal data base approach.
c. providing proof of the concept of how the environment response can be delivered to the user.

In our implementation, the Intelligent Environment is equipped with two types of sensors i.e. fixed/precise sensors and proximate sensors. The fixed sensors uses RFID sensor, door sensor, chair sensor, phone sensor, keyboard activity sensor, and mouse sensor to sense the state of user in precise location (within 1 meter). The proximate sensors uses the Bluetooth and WiFi (IEEE 802.11b,b+,g) to sense the state of user in proximate location. Both types of sensor detected current state/context by examined the aggregation sensor data and determine what actions should be taken on the given user situation.

This study is aware of the existence of a huge sensor data volume in an Intelligent Environment, with the sensor database using the fusion sensor database design for the spatio-temporal data base approach, with the partition of the database based on the sensor type and time access proposed to give reasonable time access to reuse data.

The proof of concept of how the environment response can be delivered to the user is also provided by presenting three scenarios for social environment models to enable the Active Office to respond to the user situation, i.e., when there is a meeting, the automatic login/logout, and how the environment responds to a user's phone call, that is followed by the monitoring of sensors' activities and the evaluation of the Active Office in responding to user activities.

Chapter 9

EVALUATION STRATEGY IN INTELLIGENT ENVIRONMENTS

9.1 Defining "Good Quality" Project in Context-Aware Computing

Over the past three years at least, it has become apparent that for an Intelligent Environment project to be good quality, it must be evaluated. Recently, interest in understanding evaluation problems that arise from Ubiquitous Computing is increasing the understanding of how context-aware systems are evaluated (Scholtz 2001; Dey 2002; Schmidt 2002). Evaluation strategy in the Intelligent Environment area should follow the evaluation of Context-Aware Computing, with this strategy also relevant to the evaluation of Ubiquitous/Pervasive Computing, Ambient Intelligent, Nomadic Computing and Sentient Computing (see Figure 2.1 in Chapter 2).

The multi-disciplinary area of Ubiquitous Computing combines technology, computer systems, models and reasoning, social aspects, and user support. A good quality project must have interesting *core content* in one or more of those areas, be founded on a base of scholarship, and provide an *evaluation* process. The goal is the creation of a useful, usable and elegant Ubiquitous Computing system.

9.1.1 Evaluation Process for Context-Aware Computing

The aim of the development of an evaluation program in Ubiquitous Computing is to determine what to add, how to test and the appropriate metrics to use. The development evaluation should focus effort on a specific quality research direction as well as how to instrument a computing environment for the success of the software demonstration.

Although many projects' demonstrations show only novel invention or application of technology, this is not the only way to create valuable research, and the demonstration is only a hook for the user's interest, not the whole of the project: very good invention must relate to the problems and goals specific to Ubiquitous Computing and even though it may extend them, must have some degree of evaluation.

For evaluation the Ubiquitous Computing system can be divided into subsystems, e.g.:

- Perceptual user interfaces.
- The structure of user preference.
- Data localisation.
- User mobility patterns.
- Zero configuration and dynamic service discovery.
- Wired and wireless network management.
- Distributed context processing systems.

The issue in this approach is how to establish metrics and evaluation methodology for individual subsystems and evaluation for the entire system. Considering that the computer disappears in Ubiquitous Computing, one of the possible metrics could be the amount of remaining visibility of the computer, measured by distraction[4] met by the user in completing a task.

The user's perspective in the determination of metrics is also very important; the metrics could be system workload against user workload, performance, usefulness, and usability of the end product.

To construct a realistic workload model that enables the comprehensive and fair testing of support infrastructures for Ubiquitous Computing, the workload model should have the ability to synthesise a complete range of possible application workloads that are parameterised along several dimensions. The most often used are the user and environment dimensions.

The details of both dimensions are described are follows:

- User dimension:
  - user identity/preference/personal interest.
  - user presence.
  - user location.
  - user mobility.
  - user activity.
  - user situation.
- Environment dimension:
  - Distributed sensors data fusion and processing.
  - Mobility of application.
  - Architecture of infrastructure.
  - Device capabilities.

9.1.2 Core Content for Context-Aware Computing

In a computer related project, it is common to focus mostly on computer technology aspect in the whole of the project, instead of social aspect in user dimension. However, there are other areas of equal merit that bring good combinations of areas such as integration and systems approaches. The project must have identifiable *core content*, which may be in one or more forms:

- Computer technology.
  - New hardware for sensors in old or new environments, or communication systems.
  - User interfaces.
  - Combinations of sensors, actuators, and smart sensors (filters).
  - New "devices", whether new principles or new applications for existing principles.
- Infrastructure.
  - Hardware or software toolkits, or hardware infrastructure.
  - Software infrastructure for programming with sensors; software for middleware for installing, configuring and communication.
  - System design techniques and implemented tools.
- Systems.

[4] The state of mind in which the user's attention is diverted from an original focus or interest.

    - Integration of technologies (from inside and beyond Ubiquitous Computing).
    - Installation and deployment in real environments; application of other people's novel technology (such as motes or toolkits) into systems, interactions between technologies.
- Context models and reasoning.
    - Expressive, simple but powerful.
    - Ontology.
    - Systematic models.
    - Supporting software tools.
- User support.
    - Ways and means and tools for deployment of sensors and devices, installations, productive use.
- Usability of HCI.
    - focuses on individual interaction with the context-aware system.
    - Trust and security models and analysis of specific technology or applications.
    - User centered design.
- Social aspect
    - Social application areas such as health, education, personal care.
    - Social interactions mediated by technology.

Based on those *core contents*, the social aspect and also computer technology aspect can be considered as principal criteria for "good research" in Ubiquitous Computing. Both aspects should have tight inter-relationship in design. Abowd (1999) proposes criteria for good Ubiquitous Computing research as follows:

- There should be a motivating application.
- The system built should address some notion of scale.
- The system should be subjected to real and everyday use.
- The use of the system should be evaluated to determine its impact on the user community.

The last two criteria are very important, because together they present a serious robustness challenge in an uncommon computing domain (Abowd 1999). Unfortunately Abowd did not give any weight to social issues. However, Weiser underlines, as Abowd quotes in (Abowd 1999), that evaluation and scale are relevant, but they are often ignored. The points were made to argue that people should not stop when they have built a device, but to attempt some evaluation.

In the Ubiquitous Computing area, the specific motivating applications are not always needed, as long as an application domain is known its infrastructure can be good Ubiquitous Computing research. Real and everyday use is beyond most systems, this makes the setting of the evaluation hurdle higher than necessary. This goes to Abowd's argument about software engineering for robustness of the software, which is not as generally required as he would make out in classifying "good" research.

Designing good quality research should include a whole system viewpoint which has iterative evaluation of its design. Therefore, evaluation of the design needs more careful attention of redesigning for robustness of the context-aware system, particularly when the following situation occurs, as stated by (Fithian, Iachello et al. 2003):

- Program once, run anywhere, especially for trans-rendering for different types of user terminals.

- Stateless interaction model.
- Visibility of the information needed to perform an action at any given step.
- Atomic or short interaction sequences.
- Appropriate timeout on an unfinished operation.

In relation to designing good quality research, Abowd (1999) proposed three principles that need to be included in Ubiquitous Computing, which are:

- Transparent interaction technique.
  This includes handwriting and gesture recognition, freeform pen interaction, speech, computational perception, tangible user interfaces (using physical objects to manipulate electronic information) and manipulation interfaces (embedding sensors on computational devices to allow for additional modes of interaction).
- Context awareness.
  Allowing for rapid personalisation of computing services.
- Automated capture.
  Recording everyday experiences and making that record available for later use.

Context awareness is a good goal, but there can be good ubiquitous systems that are not explicitly context **“aware”**, the ubiquitous systems may be designed to be context **“appropriate”.** The difference is that there might be *an explicit model of context that affects behaviour* or that elements of the system are deployed and configured in such ways that there is *no explicit element*, and where the behaviours in different situations are acceptable.

9.2 Evaluation Criteria for Context-Aware Computing

Evaluation in Context-Aware Computing is not seen as a once-and-for-all exercise to be carried out at the end of system development and to gain the value of general lessons learnt from the development process and the evaluation of the systems. However, a set of tools ranging from the heuristic through the empirical to the theoretical should be applied progressively throughout the software engineering development life-cycle (Downtown 1991).

In the engineering development life-cycle, *the system analysis phase*, analysis errors can be corrected by changes to user technical specification needs; *at the design phase*, errors may involve hardware and software and may invalidate design efforts which have been completed; in *the production phase*, design errors require not only a redesign of the hardware and/or software but also redesign of the tool-kit and equipment involved in the production life-cycle. Therefore, the earlier the human-factor issues in the design can be addressed, the less the cost to be incurred in correcting any errors and misconceptions (Downtown 1991) to avoid a “snowball effect”.

Context-Aware Computing performance depends on the performance of the computer system built as well as the user. Evaluation of the system as a whole must include evaluation of the user using the system. Unfortunately, the user’s responses and performance are very much less predictable that those of the computer’s system.

In the Intelligent Environment with support of the DiCPA architecture, the development context-aware application with capability to deliver service inter-domains or the federation Intelligent Environments will be possible but it still confront a very high programming complexity. Managing programming complexity is usually done by group programmers in pairs to develop context-aware applications. The incremental development approach, such as Extreme or Agile programming can be used for this purpose, requires a highly qualified and motivated staff and good working condition, otherwise software projects may be delayed because staffs are not happy about their working conditions or there is a lack of qualified staff.

The evaluation of a Context-Aware Computing environment requires *system testing,* which is based on the taxonomy (categorisation) of features for context-aware applications, to study users during regular activities and to evaluate the dependencies between the user's experience and the technologies used, including their modes of use and deployment. Example of this case, see the evaluation of the CoolTown program at HP (Spasojevic and Kindberg 2001). The finding from system testing of the user's experience and the technologies used can be used to measure the impact of the productivity and enjoyment or relaxation to the user in the context-aware environment.

System testing is one means of evaluation which uses tools that provide a methodology for verifying two aspects, i.e. *technical specification and user requirement*. One of the important technical specifications in context-aware application to be evaluated is the modelling of dynamic context resources discovery for the users in experiencing daily context to the resources at their current location in the context-aware environment. However, to achieve an objective evaluation and to avoid weak interpretation, the evaluation technique in Context-Aware Computing can use a *formal methodology* of statistical analysis with experimental psychology analysis considered as a second option.

Even though formal evaluation methodology has a weakness in that it can't cope with the wide range of variables encountered in real-life situations, the finding or result from that can be used in the understanding of the Context-Aware Computing system. Based on the understanding of the stated performance of the system, it can be improved later on. The formal evaluation can be applied by developing a testing algorithm in a crucial component. For example, the algorithm for evaluation training data-set to determine the performance in estimating user location as discussed in Section 5.8 and the algorithm is in Section 5.7.2.

Based on the descriptions above, the criteria for the evaluation of Context-Aware Computing may be considered as follows:

1. Value of general lessons learnt to stimulate development of the architecture of the hardware and software infrastructure, and user interface toolkits.
2. Impact of the productivity and enjoyment or relaxation in the context-aware environment.
3. Programming complexity to develop a context-aware application prototype.
4. Taxonomy (categorisation) of features for context-aware applications.
5. Dynamic context resources discovery modelling.

9.3 Metrics Evaluation for Context-Aware Computing

To be good quality, context-aware project must be evaluated. Unfortunately "evaluation" has different meanings to different people, Elliot Stern proposes evaluation as (Stern 2003):

- Judgment: assessing what has achieved.
- Explanation: understanding what works.

- Development: improving implementation.
- Empowerment: strengthening institutions, communities and networks.

Even though Stern's definition was for the evaluation of education, social and organisational programs, his definition is embedded in computer systems as well.

In the computing area evaluation has two basic approaches, *formative evaluation* and *summative evaluation* (Scriven 1967). *Formative evaluation* is evaluation done during development to improve a design, and summative evaluation is evaluation done after development to assess whether or not a design fulfils a certain set of criteria. The set of criteria can make a better tool, it may relate to a software engineering approach in efficiency and software testing, or it may relate to usability issues to meet the requirement's specification. *Summative evaluation* is used to compare the level of usability achieved in an interaction design. This evaluation is generally regarded as rigorous, formal experiment design, includes a test for statistical significance and is often used to compare design factors to accumulate knowledge in the field of study (Hartson, Andre et al. 2003). From these basic approaches, the quantitative measurement of product and process in software development can be derived metrics evaluation for software quality in Context-Aware Computing fields.

The beginning of the evaluation of software quality was modelled in the 1970s, mostly in theoretical models with Knuth's proving correctness of a small program using a mathematical approach (Knuth 1997). The Boehm model (Boehm, Brown et al. 1976) and McCall Model (Cavano and McCall 1978) are the early software quality models that are closest to the evaluation of good quality research in the area of Ubiquitous Computing. Both models focus on the software product and not on the analysis and design processes. The models have "software metrics"[5] which define basic user needs, quality factors and quality attributes (Table 9.1). The quality factor of the product is the highest level quality attribute, and is decomposed into quality criteria and quality attributes with associated metrics.

When a user purchases new software or a new device, the user will have questions such as: Is the product easy to use in its present state? How easily can the software/device product be modified? How will it be possible to use the software/device product in another hardware/software environment? In reflecting on these questions, the Boehm and McCall Models provide three basic principles:

1. The usability of the product in its present state ("as-is utility" in the Boehm model and "product operation" in the McCall model).
2. The changeability of the product's functions ("maintainability" in the Boehm model and "product revision" in the McCall model).
3. The portability of the product into another hardware-software environment ("portability" in the Boehm model and "product transition" in the McCall model).

The developer and the user can collaborate to define the criteria and attributes of metrics product evaluation. Boehm and McCall models show several quality factors/characteristics with six of them the same, i.e., portability, reliability efficiency, usability, testability, flexibility (shown in bold in the Table 9.1).

[5] The factors or the attributes characteristic of the product or process which can be measured numerically in the software development.

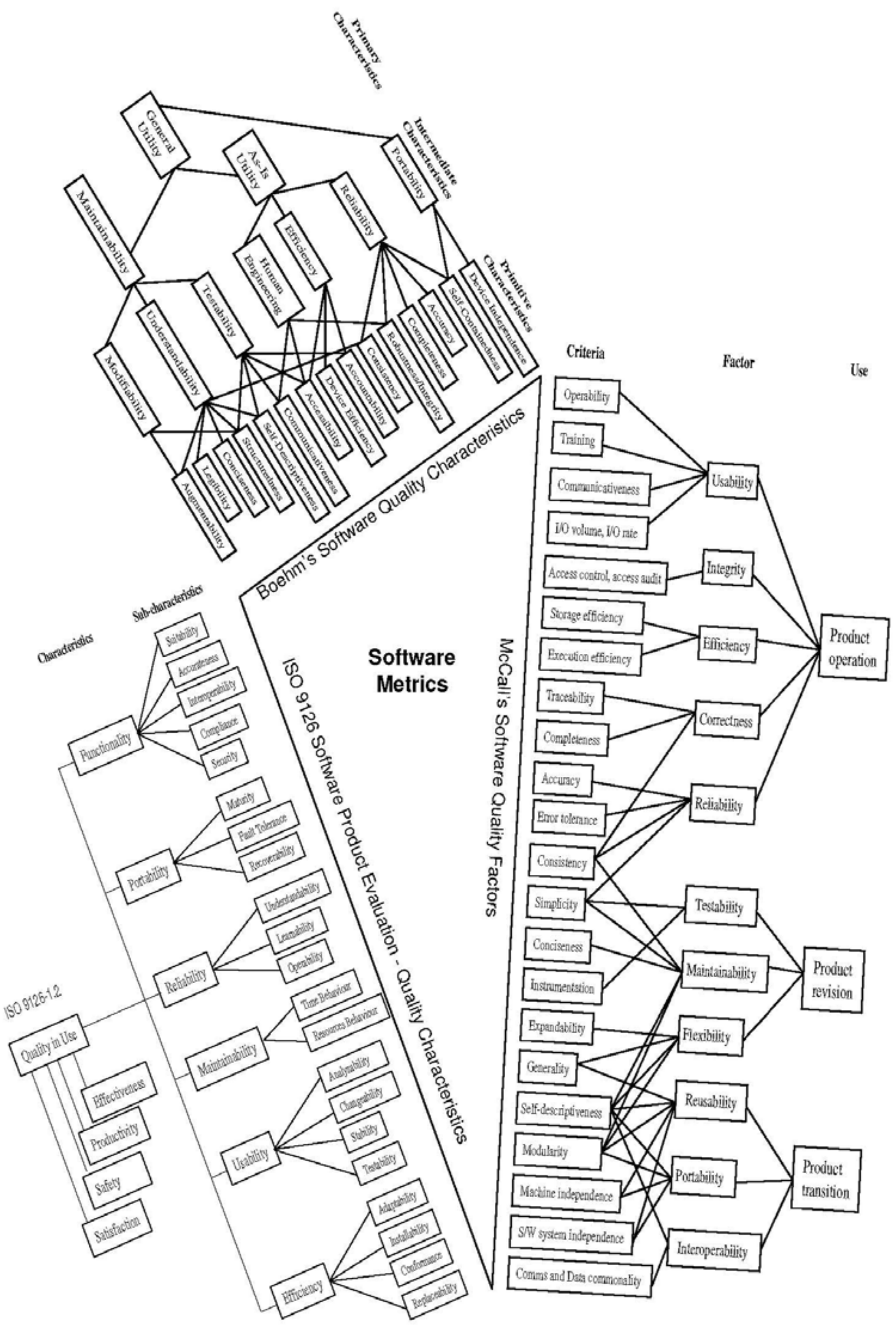


Figure 9.1 Software Quality Metrics of Boehm Model, McCall's Model and ISO/IEC 9126

Table 9.1 Software Quality Metrics of Boehm Model and McCall Model

	Boehm Model	**McCall Model**
Basic user requirements	1. "as-is" utility 2. general usability 3. maintainability	1. product operation 2. product revision 3. product transition
Quality factors/characteristics	1. **portability** **2. reliability** **3. efficiency** **4. usability,** 5. **testability** 6. understandability 7. **flexibility**	**1. usability** 2. integrity **3. efficiency** 4. correctness **5. reliability** 6. maintainability 7. **testability** **8. flexibility** 9. reusability **10. portability** 11. interoperability
Sub-characteristics/metrics	1. Device Independence 2. Self-Containedness 3. Accuracy 4. Completeness 5. Robustness/Integrity 6. Consistency 7. Accountability 8. Device Efficiency 9. Accessibility 10. Communicativeness 11. Self-Descriptiveness 12. Structuredness 13. Conciseness 14. Legibility 15. Augmentability	1. Operability 2. Training 3. Communicativeness 4. I/O volume, I/O rate 5. Access control, access audit 6. Storage efficiency 7. Execution efficiency 8. Traceability 9. Completeness 10. Accuracy 11. Error tolerance 12. Consistency 13. Simplicity 14. Conciseness 15. Instrumentation 16. Expandability 17. Generality 18. Self-descriptiveness 19. Modularity 20. Machine independence 21. SW system independence 22. Comms and Data commonality

Software Product Evaluation of ISO/IEC 9126 (1991) can be seen as the next generation software quality evaluation framework from the Boehm model and the McCall model, providing a framework for the evaluation of software quality. The comparison metrics of the two models are described in Figure 9.1.

The software product quality of ISO/IEC 9126 defines a quality model which is applicable to every kind of software (ISO/IEC-9126 1991), but which does not provide requirements for software. It recommends six product quality characteristics and the annex provides a suggestion of quality sub-characteristics. ISO/IEC 9126 has a complete set of evaluations which are relevant to software product quality of Context-Aware Computing. The new interpretation details of *software product quality metrics for Context-Aware Computing*, based on Boehm Model, McCall Model and ISO/IEC 9126 with some additional or modification of metrics and interpretation are shown in Table 9.2. This new interpretations are based on two aspects of the *evaluation process*, which is the technological (Section 9.1.1) and the social aspects and two dimensions of the *core content*, which is the technological and social dimensions (Section 9.1.2).

Table 9.2 Software Product Quality Metrics for Context-Aware Computing

1. Functionality	• **Suitability**	The presence and appropriateness of a set of functions for specified instance context tasks
	• **Accurateness**	The provision of right or agreed results or effects
	• **Interoperability**	The ability to interact with specified familiar and unfamiliar environment computing systems.
	• **Seamless**	the degree of consistency and coherence required to continue operation across changing circumstances for mobile user.
	• **Compliance**	Making the software adhere to application related standards or conventions or regulations in laws and similar prescriptions.
	• **Security**	The ability to prevent unauthorised access, whether accidental or deliberate, to programs or data, in a seamless manner.
	• **Responsive**	The ability of the computing environment to respond to the activity of the (authorised) user.
	• **Assistance**	the ability contribute to the fulfilment of a need or furtherance of an effort or purpose
2. Reliability	• **Maturity**	The frequency of failure by faults in the software or hardware devices.
	• **Fault tolerance**	The ability to maintain a specified level of performance in the case of software and devices faults, or of infringement of its specified interfaces.
	• **Scalability**	The capability to give response in accord with a particular proportion or scale in a computing environment
	• **Recoverability**	The capability to re-establish its level of performance and recover the data directly affected in case of a failure and on the time and effort needed for it.
	• **Transparent**	The capability to be seen or understood particularly through a small device in transmitting service in the computing environment
3. Usability	• **Understandability**	The users' efforts in recognising the logical concept and its applicability especially when the user is in instant context; the awareness of the users that they are in instant context between them and with the computing environment.
	• **Learnability**	The users' efforts in learning its application and its context applications.
	• **Operability**	The users' efforts in operation and operation control.
4. Efficiency	• **Time behaviour**	The response and processing times and on-throughput rates in performance functions.
	• **Resource behaviour**	Attributes of software that bear on the amount of resource used and the duration of such use in performing functions.
5. Maintainability	• **Analysability**	The effort needed for diagnosis of deficiencies or causes of failures, or for identification of parts to be modified.
	• **Changeability**	The effort needed for modification, fault removal or for environmental change.
	• **Stability**	The risk of unexpected effects of modifications.
	• **Testability**	The effort needed for validating the modified software and

		devices.
6. Portability	• **Adaptability**	The quality of being adaptable or flexible in different situation in specified environments without applying actions or means other than those provided for that purpose for the software and devices considered.
	• **Sensitivity**	The capability to be responsive to changes of condition in a rapidly changing computing environment
	• **Installability**	The effort needed to install and set up the software and devices in a specified computing environment.
	• **Conformance**	Making the software and devices adhere to standards or conventions relating to portability.
	• **Replaceability**	The opportunity and effort of using it in the place of other specified software and devices in the computing environment of that software and device.

The quality factors or characteristics of “flexibility” which appears only in the Boehm and McCall Model are relevant and have new interpretation as “adaptability” in context-aware computing metrics evaluation from the ISO/IEC 9126. The quality factors or characteristics of “consistency” which appears only in the Boehm and McCall Model are relevant and have new interpretation as “seamless” in context-aware computing metrics evaluation, but do not appears in the ISO/IEC 9126. The other metrics which are included and not a part of the three models above are transparent, responsive, assistance, sensitivity and scalability.

Further, as shown in Figure 9.1, ISO/IEC 9126-1.2 facilitates further development of software quality products and introduces the concept of “quality in use”. In relevance to Context-Aware Computing evaluation, based on ISO/IEC 9126, the new interpretation requires for example the development of the user's view of Context-Aware Computing quality and characteristics, as well as the user’s view of the computing environment ornaments including software, sensors and devices. The Context-Aware Computing evaluation concept also proposes to measure results by using the software in the environment, rather than the properties of the software itself.

Quality in the user's environment may be different from the developer's environment, because some functions may not be visible to a user, may not be used by a user, or because of the properties of the user’s location (crowded areas or an area with bad connectivity).

The most important metrics in the evaluation of user using the system or user experience are the metrics of usability evaluation. These metrics bring “quality in use” issues, and will be discussed in more detail, including usability background, in the next section.

9.4 Usability Evaluation for Context-Aware Computing

By the early 1980s, the term “usability” was introduced to replace the term “user friendly” which had undesirably vague and subjective connotations. However, in the intervening years, there was no accepted definition of the term “usability” (Bevan, Kirakowskib et al. 1991).

The ISO standard for a software quality definition of “usability” from the viewpoint of product and user orientation (ISO/IEC-9126 1991) is:

> *"a set of attributes that bear on the effort needed to use, and on the individual assessment of such use, by a stated or implied set of users"*

The definition of "usability", from the viewpoint of easy-of-use and user attitude to the system (Preece, Rogers et al. 1994), is:

> *"a measure of the ease with which a system can be learned or used, its safety, effectiveness and efficiency, and the attitude of its users towards it".*

The definition of "usability" of the ESPRIT MUSiC project (Bevan, Kirakowskib et al. 1991) from the viewpoint of ease of use and acceptability is:

> *"the ease of use and acceptability of a system or product for a particular class of users carrying out specific tasks in a specific environment; where 'ease of use' affects user performance and satisfaction, and 'acceptability' affects whether or not the product is used".*

Usability has been defined in many literature sources and has been used in the field of software evaluation, but the definition did not include a social aspect. Usability minimally has two important functions that it can bring to Context-Aware Computing i.e., a technological aspect, "easy-of-use", and a social aspect, "acceptability". The technological aspects measure how capable the users are at using the system, which are the objective performance measures. The social aspects assess how much the users like the system, which is the subjective user preference.

The function of ease-of-use, including learnability when relevant, is to determine whether a product can be used. Ease-of-use in a particular context is determined by the software and device product attributes, and is measured by user performance and satisfaction.

The acceptability of a software product in Context-Aware Computing determines whether it will be used and how it will be used and determines actual usage by a particular user for a particular task in a particular social context. The context consists of the user role (user, preference, profile, task or activity) and the environment role (physical and social environment).

This approach leads to this writer's definition of "usability" in the area of Context-Aware Computing:

> *The degree of the easiness to learn, use and operate, and the comfortability of users while presence in the specified computing environment and the acceptability of users in using the software, and devices in social context.*

Usability evaluation for Context-Aware Computing has to consider the understandability, learnability and operability of the user to use the product/device in the computing environment and as well as the social environment. Usability evaluation for Context-Aware Computing can be divided into four view categories:

- the user performance view: that usability can be measured by examining the user's effort to interact with the software or device product for operation, control operation, input and output processes.

- the communication view: that usability can be measured in terms of the user's effort, his mental effort and the attitude of the user in communicating with the product/device.
- the user-understandability view: that usability can be measured in terms of the user's effort in understanding the consistency, structuredness, traceability, self-descriptiveness, conciseness and legibility of the product/device.
- the product-oriented view: that usability can be measured in terms of the ergonomic attributes, expandability and independence attributes of the product and communication between users, products and network environments.

9.5 The Evaluation of this Work

This section will discuss the evaluation of this study. The evaluation of the *social aspect* and *computer technology aspect* as principal criteria of *core content* of this study in Context-Aware Computing area is presented (Table 9.3). Some prototype demonstrations are evaluated in case by case studies, by using several evaluation metrics, such as *scalability*, *analysability* and *stability*. The evaluation and system testing for verifying technical specifications using formal evaluation methodology are discussed. Finally, the value of general lessons learnt from the prototypes of this work is also described.

The *core content* that can be identified from this study would be involved as follows:

- Computer technology: 1. User interfaces. 2. Combinations of sensors, actuators, and smart sensors (filters). 3. New principles for evaluation.
- Infrastructure: 1. Hardware or software toolkits, or hardware infrastructure. 2. Software infrastructure for programming with sensors; software for middleware for installing, configuring and communication. 3. System design techniques and implemented tools.
- Systems: 1. Integration of technologies (from inside and beyond Ubiquitous Computing), such as machine learning, spatio-temporal databases and fusing sensors data. 2. Installation and deployment in real environments in an Active Office, interactions between technologies.
- Context models and reasoning: 1. Expressive, simple but powerful such as ηk-Nearest Neighbour (ηk-NN) Algorithm. 2. Systematic models.
- User support: 1. Ways and means and tools for deployment of sensors and devices, installations, productive use.
- Usability: 1. focuses on individual interaction with the context-aware system. 2. User centred design.
- Social aspect: 1. Social interactions mediated by technology, such as scenario of "Having a guest" described in Sections 7.5 and Modelling Social Environments in Responding to User Situations described in Section 8.6.

9.5.1 Core Content of This Work

A good quality project must have one or more elements as core content, and as stated in Section 9.1.1 of this work the core content includes: a user location model, user mobility model, user activity model and the response of the environment model to assist and help the user. The last core content to make these models synchronous and integrated as a system is DiCPA architecture, which provides a single map of the system structure for distributed context processing to work in an integrated way.

As a complete Intelligent Environment framework, DiCPA is not fully implemented in our Active Office. However, DiCPA can therefore be evaluated by design inspection analysis, whereas the others can be evaluated in the basis of the implementation studies reported here.

9.5.2 The Evaluation of Location Scalability

Scalability is needed for widespread ubiquity, therefore the *scalability metrics* of the quality factor in context-aware computing metrics is used to evaluate the design but no numerical measurements of particular proportion scalability are made at this stage of prototype development.

Location scalability, as discussed in Section 4.6, describes the structure of location information, object location of the user, devices and sensors, and how the location-aware application responds to the query that implied the user location.

In an Active Office, the location scalability may not be significant when the location hierarchy is defined only as a single level, which is because of:

- The scope of the query will also be in the single level and need not consider another building or enterprise of the Active Office
- Most context-location information will only require local information
- Location properties/attributes and location authority/location reference can be considered complex because of many ways to refer to rooms, devices and sensors; however, using a single level structure, the location information will be simple to be referred/used.

In location scalability area of study, the degree of capability in sensing user location may not be similar from one office to another. There may be different methods and different proximate and precise sensors. This would be an important consideration before implementing the location scalability at more than one level.

However, when an Active Office designes with more than one level of location scalability, by changing the query and *location hierarchy*, the response from the Active Office can be extended, for example change in the queries from building level to university level, can expand the scope of the query and as a result it will expand the response in answering the query.

The scalability of object location in responding user to a location query in an Active Office, as describe in Section 4.6, depends on the *"location hierarchy"*, *"input query", "request query" and "scope query"* which could provide proper but scalable location information of user location query.

The large scale of location information, i.e. the large number of location, in some cases, such as in a campus, across many institutions, or in the world, may not necessarily be an issue, as long as the infrastructure system provide clear wide scalability of the *"location (naming) hierarchy"* and the user query has clear *"input query", "request query" and "scope query"* structure. Perhaps, in future context-aware environments, if the provision of instrumentation and information can be combined, location scalability straight away can follow.

Table 9.3 The Evaluation of the Social/Computer Technology Aspects and User/Environment Dimensions of this Study in Context-Aware Computing

Model/Prototype/ Scenario	Source (Section)	Type of Evaluation	Method of Evaluation	Measurement	Core Content/ Software Quality Metrics	Outcomes or issues
Partition of the spatio-temporal database.	- 9.5.3	- Technology aspect.	- Proof of Concept. - Proof of performance.	Quantitative Measurement (measure sensors data of 2 users in 5 working days).	- Scalability metric. - Selectivity metric.	- Handling the very quick grow of sensor data in responding user query. - Partitioning of the spatio-temporal database based on time and sensor.
Situation for automatic login/logout.	- 8.6.2 - 9.5.5 - a	- User dimension.	- Proof of performance. - Modelling social environment.	Quantitative Measurement (measure the size of 93 user profile are observed).	- Social aspect metric. - User support metric. - Responsive metric.	- Disturbing user (when the system has a slow response). - How to produce a reasonable response (for instance, the profile max 20 Mbyte) - Unexpected scenario is discussed.
The chair's sensor scenario.	- 9.5.5 - b	- User dimension.	- Proof of performance. - Modelling social environment.	No quantitative Measurement.	- Social aspect metric. - User support metric.	- Disturbing user (jitters make slow response). - How to get proper response by normalising sensor data. - Unexpected scenario is discussed.
Situation to respond a phone call (*Ir*Media Player).	- 8.6.3 - 9.5.5 - c	- User dimension.	- Proof of performance. - Modelling social environment.	No quantitative Measurement.	- Social aspect metric. - User support metric. - Operability metric.	- This prototype capable to automatically reduce/pause a multi media volume as a response when user get a phone by avoiding inappropriate background sound in the environment - To get proper response, the *Ir*Media Player needs user identity (Bluetooth enabled mobile phone).

						- Unexpected scenario is discussed.
Response to user weaknesses in Active Office.	- 8.6.2 - 9.5.5 - d	- User dimension	- Proof of performance. - Modelling social environment	No quantitative Measurement.	- Social aspect metric. - User support metric.	- A user forgets to take his mobile phone when leaving office. - A user disturbs other user - Unexpected scenario is discussed.
A Meeting scenario.	- 8.6.1	- Technology aspect.	- Proof of concept.	No quantitative Measurement.	- Combination of sensors. - Integrated Technology.	- Capability to understand the activity and the situation in the meeting. - Capability to respond when there is disturbance from outside.
Location Scalability Model.	- 4.6 - 9.5.2	- Technology aspect. - Environment dimension.	- Proof of concept.	No quantitative Measurement.	- Scalability metric. - Sensitivity metric.	- Location hierarchy (structure of location information, object location of user, devices and sensors).
Sensor Activity Monitoring system.	- 8.7	- Technology aspect.	- Proof of performance.	Quantitative Measurement (measure 11 sensors, 7 access points and 2 Bluetooth scanners.)	- Combination of sensors. - Integrated Technology.	- Monitoring sensor's activities graph. The dynamic web-based graph to monitor precise and proximate sensors activity
Speech Context Agent (SpeechCA) for communicate a user location.	- 4.5	- Technology aspect.	- Proof of Concept. - Proof of performance.	No quantitative Measurement.	- Integrated Technology. - Interoperability metric.	- Capable to capture user instruction and dictate response.
Mobile Access Point.	- 7.5	- Technology aspect.	- Proof of Concept. - Proof of existence.	No quantitative Measurement.	- Combination of sensors. - Integrated Technology.	- The access point which is mobile for a guest user. - Capable to provide public access to multiple guess users in their unfamiliar domain. - Capable to estimate a guess user location.

Precise Sensors System.	- 4.3.1	- Technology aspect.	- Proof of performance.	Quantitative Measurement (measure 3 users for 3 months working days).	- Combination of sensors. - Integrated Technology.	- In a normal situation, precise sensors system capable to get 100% correctness in capturing user location.
Proximate Sensors System.	- 4.3.2	- Technology aspect.	- Proof of performance.	Quantitative Measurement (measure 2 users' activities in a working day).	- Combination of sensors. - Integrated Technology. - Accurateness metric.	- It used self organizing map and then verified using *ηk*-Nearest Neighbour Algorithm, the result is almost the same but it is faster and simpler. - Accurate estimates (95.82%) in rooms of 3 meters width with minimal access points
ηk-Nearest Neighbour Algorithm.	- 5.5 - 5.7.1 - 5.7.3	- Technology aspect.	- Proof of existence. - Proof of performance.	Quantitative Measurement (measure signal strength and signal quality from 7 WiFi's access point for a user in every 14 hours) continuously for 11 days.	- Combination of sensors. - Integrated Technology. - Accurateness metric. - Responsive metric.	- The use of instance-based learning method to classify and estimate user location. - This algorithm is also compared with 3 other K-Nearest Neighbour and self organising map, and found as the best and stable result. - The result depends on the quality of training data-set and the variety of the sensors. - Classify two kinds of zones: noise zone and stable zone in estimation of user location.
Boolean MaxMin Algorithm.	- 5.6 - 5.7.2 - 5.7.3	- Technology aspect.	- Proof of existence. - Proof of performance.	Quantitative Measurement (measure signal strength and signal quality from 7 WiFi's access point for a meter point, 6	- Combination of sensors. - Integrated Technology. - Fault tolerant metric. - Accurateness metric.	- Determining the quality of training data-set for *ηk*-Nearest Neighbour. - Finding the importance of data normalisation and the maximum common value (best k) to achieve the maximum correct result in

				times each for 4 rooms and 3 corridors in a building).		estimation of symbolic user location.
Distributed Context Processing Architecture (DiCPA).	- 3.1 - 3.3	- Technology aspect.	- Proof of existence.	No quantitative Measurement.	- Adaptability metric. - Replaceability metric. - Scalability metric. - Transparent metric. - Analysability metric.	- A scalable context processing architecture with the capability of delivering service while the user is moving in an Intelligent Environment. - Designed to provide the interactive and interaction process between user and the environment. - DiCPA has four context layers architecture to enable the system to recognise user activity. - Comparison with other architecture, such as IROS and Aura is discussed.

9.5.3 Advantage and Weakness in the Partition of the Spatio-Temporal Database

The growth of sensor data in a Pervasive Computing Environment cannot be avoided. As mentioned earlier in Section 8.4.1.2, it was proposed to partition the spatio-temporal database based on time and sensors. Partitioning the spatio-temporal database to manage the growth of sensor data is evaluated as another aspect of scalability and selectivity as to how a context-aware application chooses the relevant sensors. This is related to scaling in locations, which is represented by the subsets of the sensors that are associated with locations.

The advantage in the partitioning of the spatio-temporal database based on time is that the application will use the smallest database to get the fastest response, but the weakness of this approach is:

- If the number of sensors grows very large, the spatio-temporal data will also grow very large.
- Even though the filter database is limited by time, it still has the potential to grow very large. The longer the filters are in place the greater the data.
- The history database (SensorDB) is a database that keeps all data without any filter. It has the potential to grow very large in a short time, and in consequence have a slow response to the application.
- When the application uses only a small number of sensors, for example 3 sensors out of 100 sensors, it has to search the whole (100) sensors.

The advantages of the partitioning of the spatio-temporal database based on sensors are

- the database will not grow very large as is the case in the partition above based on time.
- no data redundancies.
- the application can access based only on sensors that it needs, e.g. when the application needs 3 sensors out of 100 sensors, it will only access three sensor's databases.

The weakness of the approach is that the number of the sensor databases will also grow as the number of sensors grows. However, this approach is good for the environment that uses a fixed number of sensors.

The best option can be the combination of the above approaches; 1. to use the partitioned spatio-temporal database based on sensors and 2. for the partitioned sensor data that grows very quickly, which is from sensors that are used very often, the partitioned spatio-temporal database is partitioned again based on time.

In our experiment, the sizes of sensors data fluctuate significantly. For example, the average size of sensors data is between 43,358 byte for keyboard sensor and 140,154,720 byte for WiFi sensor, both in plain text data, for the observation of 2 users in 5 working days. It reached between 3.165 Mbyte and 10.231 Gbyte plain text data per year. To increase the performance of user query to the location sensors system, the keyboard sensor data, as it does not grow quickly, is kept in the history database (SensorDB) and the WiFi sensor data, as it grows very quickly and it is used very often for user location purpose, is partitioned separately based on sensor and time.

9.5.4 Evaluation of the Sensor's Activity

This section discusses the use of *analysability* and *stability* metrics from Table 9.2 by looking at the sensor's activity. This evaluation is based on real experience and the observation of the prototype without any quantitative measurement. When there exists any association between a user and the proximate and precise sensors, the *analysability metric* diagnose the deficiency or cause of failures and *stability metric* gives attention to unexpected effects of changing state.

The purpose of evaluating the sensor's activity is to monitor the performance of the sensor while sensing the objects and delivering a response to the user. In this study, the graph for all sensors, as shown at Section 8.7, monitors the behaviour of sensor activities, who uses the sensor, and the time and status of the sensor, can also be shown. When a sensor does not work properly, by diagnosing the history data from the graph, the cause of failures can be identified and analysed. The *analysability metric* can be used to measure the effort from identifies and analyse the malfunction of the sensor that needs to be modified, not the measurement of the frequency of the failure.

Stability metric can be measured by how reliable or stable is a sensor data that needs to be filtered or smoothed, for instance, when a sensor has a lot of jitter (such as pressure chair sensor point B in Figure 8.7) a lot of data which is recorded may not be useful enough for the Intelligent Environment. Jitter cannot be avoided, but it can be reduced.

The jitter on a pressure chair sensor, for example, happens when the user is sitting on the chair, which cannot be avoided, because a user always moves anytime and anywhere whether he is aware of this or not, as a part of normal human behaviour. The jitter can be reduced by finding a better sensor for user movement absorption or by developing a technique to normalise the collection of the sensor data and to study the pattern of user movement on that chair, before it is stored to the database.

9.5.5 Evaluation on the Modelling of the Social Environment

For the purposes of the evaluation of the use of the sensor's activity in the modelling of the social environment, four cases presented in Section 8.6 will be discussed:

a. The situation when an Active Office responds to user login automatically
b. Problems with the chair's sensor
c. The situation when an Active Office responds to a phone call
d. The situation where a user is off his chair for more than 10 minutes

Evaluation of context-aware prototypes, as mentioned earlier in this chapter, can be evaluated from the *user dimensions*, such as user activity, and the *social aspect* which is social interactions mediated by technology, including *user support*, which comprises ways and means and tools for deployment of sensors and devices for productive use from case four above. The evaluation of the social issues will consider 1. how the system may be disturbing for the user, such as slow response, 2. how to get proper response from the computer environment, or, 3. a weakness, an unexpected scenario, the user may forget to take his mobile phone when leaving the office, may making an incorrect interpretation of the environment. However, the solutions to the problem scenarios above, are also discussed.

a. The situation when an Active Office responds to user login automatically

When a user comes into his room in an Active Office, his desktop computer will normally automatically login for him after he sits on his chair, on the condition that he turns on his mobile phone with a Bluetooth enabled device in discoverable mode before he sits. If he sits on the chair but has forgotten to enable Bluetooth discoverable mode, the Active Office will not give an appropriate response.

In this case, the user has to unlock or login his computer manually, or get up from his chair, turn on his mobile with Bluetooth enabled device in discoverable mode and then sit.

In the automatic user login scenario, a user may follow a valid sequence of steps, an appropriate response may occur, but in some cases, the user still feels inconvenience. For example, when a user has a large profile in his desktop computer, the user comes and goes frequently from his room with his Mobile Phone in a short period of time, the desktop computer will respond very slowly, which is of course, disturbing for the user. In this study, 93 user profiles were observed and the range of the profiles was between 420 Kbyte – 1.2 Gbyte, and from the experiment to produce a reasonable response, the size of the profile may not exceed to 20 Mbyte. Hence, the large profile needs to be reduced to have an acceptable response.

b. Problems with the Chair’s Sensor

A chair’s sensor is very sensitive. Human user/occupants are always moving while sitting. The chair’s sensor can frequently change state in a very short time (jitter) but the record of every changed state is added, creating problems for the database as the number of records can be dramatically increased.

In this case, two approaches may solve the problem. First, by finding a better sensor for a user that absorbs movement and reports fewer but more significant events or, second, by normalising the sensor data by studying the movement patterns of the user when on the chair. If the database is normalised, there is potential for it to be in conflict with the case in point a above, such as in (Brumit, Meyer et al. 2000). This is an important point for the system design, such as, where one scenario is a waiting state for sensor data to deliver an appropriate response, another scenario may eliminate the expected sensor data.

c. The Situation Where an Active Office Responds to a Phone Call

In the scenario of respond to a phone call, in Section 8.6.3, when a user has a phone call through a regular fixed lines phone, a normal response from an Active Office is to reduce or turn off the volume of his computer’s sound or other small device while he is on the phone. Unfortunately his Bluetooth enabled mobile phone in discoverable mode is needed to represent his user identity to deliver the response to reduce or turn off any volume on his computer. This means that to get the proper response in receiving a call on a regular fixed line phone, the user needs his Bluetooth enabled mobile phone that can be used as user identity.

This context aware application can be expanded to other social situations, for example, where a user has an extended wireless phone from fixed line phone, it can be designed to get the same response from the scenario above, to reduce or turn off any volume in his computing environment.

d. The Situation Where a User is Off His Chair for More Than 10 Minutes

As mentioned earlier in Section 8.6.2, in an Active Office, when a user gets off his chair for more than 10 minutes, the desktop computer will be locked. However, in the case his Bluetooth enabled mobile phone is still in his office, the desktop computer will remain active for the user to work. This is one of the weaknesses; if a user forgets to take his mobile phone home when leaving the office, his workstation will remain active.

Another problem is where a user disturbs other users by the use of Bluetooth technology, such as Bluejacking to a Bluetooth enabled mobile phone. It may extend not only to send text message but also send a service/agent/event to the other Active Office's user. However, there are good sides of Bluetooth technology, such as an automatic volume control as explained in Section 8.6.3 or Bluetooth image viewer technology. This Bluetooth technology views and shares picture images from a mobile phone or PDA enabled camera using "push" technology to any display, such as a TV screen or video projector. Some security issues may be raised but it is not expected to be a big problem to secure this Bluetooth control protocol in the application layer.

9.6 Summary

In the area of Context-Aware Computing, the general lack of understanding of the capabilities and limitations of the evaluation process has led to an intense need for researchers and others to be able to determine which methods are effective in what ways and to what purposes.

The evaluation of the Context-Aware Computing system, in particular Ubiquitous Computing systems, has many methods that are not standard and the lack of borrowing from other fields is unsatisfactory, making Context-Aware Computing systems lack a systemic user-centred evaluation methodology (Scholtz 2001; Schmidt 2002). Some publications use a sub-part of a system which is evaluated using well known methods, but where the complete system is not evaluated owing to the lack of a standard method. Further, evaluation methods are difficult to compare because of the lack of standard definitions, measures, and metrics on which to base the criteria for usability evaluation and comparison. Since there is no single standard for direct evaluation comparison, the result is a multiplicity of different measurements used in the studies, all capturing different data in different ways. Consequently, very few studies clearly identify the target criteria against which to measure the success of usability evaluations being examined. As a result, evaluation comparison studies are either inaccurate, or reported as incomplete or otherwise falling short of requisite scientific standards (Gray and Salzman 1998; Hartson, Andre et al. 2003). Therefore, evaluation methods for Ubiquitous Computing need to be further developed.

This chapter proposes that the "good quality" project for Context-Aware Computing requires *core content* (Section 9.1.2) and provides iterative *evaluation* processes (Section

9.1.1), which make it possible to have two types of iteration: design evaluation and product evaluation. The major iterative design process can be in three stages: initial design, prototype and final design, with iterative product evaluation following product development life-cycle, which is introduction, product growth, mature stage, saturated stage, decline stage and vanish stage (Section 2.12.3).

The *software product quality metrics for Context-Aware Computing*, based on Boehm Model, McCall Model and ISO/IEC 9126 with some additional or modified metrics and interpretation, is proposed by considering two aspects, the technological and the social aspects, and two dimensions, the user and environment dimension, as shown in Table 9.2.

There are many ways of performing the evaluation process, from systems analysis through final product, including evaluating the usability of an interaction design. Beyond this level of evaluation approach, there is still much room for disagreement and discussion about the relative merits of various evaluation techniques.

Chapter 10

CONCLUSIONS AND FUTURE RESEARCH

This thesis searched for ways in the Intelligent Environment to make Ubiquitous/Pervasive Computing work better for users by creating and equipping a computing environment. An implementation model of an Intelligent Environment is developed and called an Active Office. It is a normal office, which consists of several normal rooms with minimal intrusive detectors and sensors, and without explicitly badging people. The Active Office uses wireless communication, i.e. Bluetooth and WiFi, to enable user mobility and proximate and precise sensors to detect user location, user activity and to respond to the user when the environment detects that the user needs assistance, guide, help, or other.

The thesis addresses issues in the Intelligent Environment, particularly how to enable transparent, distributed computing to allow continued operation in a seamless manner across a changing circumstance: how to exploit the changing environment so that it is aware of the context of the user's location, the collection of nearby people and objects, accessible devices and changes to those objects over time. In order to respond to this issue, this thesis presents proofs from two directions, the '*proof-of-concept'* and the '*proof-of-performance',* which are described below.

10.1 The 'Proof of Concept'

The '*proof of concept'* is achieved by the development of the concept of:

- A scalable distribution context processing (DiCPA) architecture. The Intelligent Environment is an environment with rapid and rich computing processing. DiCPA architecture was developed to manage and respond to rapidly changing aggregation of sensor data. This architecture is a scalable distributed context processing architecture with the capability of delivering service while the user moves and it provides continued operation across changing circumstances for users over time in the Intelligent Environment. This approach focuses on how the Intelligent Environment provides context information for user location, user mobility and user activity model.
- The estimation of symbolic user location using instance-based learning method in machine learning. This concept deals with the problem of WiFi signals that fluctuate greatly, not only across perturbations in space but also in time (diurnally), which obviously leads to poor location estimation, but in which symbolic user location can be estimated accurately using instance-based learning methods. This concept also answers the "*where*" questions in context-aware mechanism in indoor environments such as in an Active Office.
- The user mobility model. This concept proposed a user mobility pattern based on changing user location (spatio-temporal database history) in an Active Office. This model can recognise the *pattern of accuracy* and the *regularity* of daily user movement and user activity.

- The user activity model. This concept proposed user activity as any association between a user and smart sensors in the environment, or any sensors being in active use to access resources. This model provided four user activity categories, i.e., an activity as association between a user and smart sensors in the environment, as a node in a work flow or job breakdown, as a physical movement mode or state, or as a mode of a state of human intent. Human activity is very complex and the computer environment is very limited in capturing user activity, but user activity is an essential component in the determination of appropriate response/assistant behaviours to these activities in order to provide appropriate services without explicit commands. This leads to a situation-based approach in Context-Aware Computing.
- Intelligent Environment response modelling. This concept recognises user activity and the capability of the environment to deliver responses. This makes an Active Office able to detect the users' current state/context and determine what responses to take based on the user's context.
- The evaluation strategy concept for an Intelligent Environment.

10.2 The 'Proof of Performance'

This thesis demonstrates the '*proof of performance'* through the development of prototypes as follows:

- A precise sensors system, a prototype application to detect user location from chair pressure sensor, magnetic phone sensor, magnetic door sensor, RFID sensor, keyboard activity sensor and/or mouse activity sensor. In normal situations, this precise sensors system is able to get 100% correctness in capturing user location.
- A proximate sensors system, a prototype application to detect user location using IEEE 802.11b,b+,g and/or Bluetooth detection. This proximate sensors system could achieve accurate estimates (95.82%) in rooms of 3 meters width with minimal access points using 2 access points data instead of the 7 access points available in the building.
- SpeechCA, a prototype application using speech recognition for the interactive query of user location. This speech prototype is able to capture user instruction from user speech and give a response by delivering action in two ways; first in the form of *guidance,* the Intelligent Environment provides step by step procedure/information for a certain purpose and the user will do the physical action; or secondly in the form of *assist/help,* which means the Intelligent Environment will deliver a response to a user request with the physical response done by the Intelligent Environment.
- *ir*Media player, the prototype application to deliver interactive responses for a media player while monitoring phone activity. This prototype application is able to automatically reduce/pause a multimedia volume as a response when user receives a phone call, avoiding inappropriate background sound in the environment.
- Monitoring sensor activities graph, the web-based graph to monitor precise and proximate sensors activity.

- A mobile access point for a guest, a prototype application of the access point which is mobile for a user in the Active Office area, but does not belong exclusively to any local Active Office domain regular user. This prototype is able to provide public access to the guest users in an unfamiliar domain and is also able to estimate a guess user location.

10.3 Future Research

This section will describe future studies which may arise from this work, and which are derived from Chapter 3,4,5,6 and 8 in this thesis.

The requirement for scalable context processing in an Active Office using DiCPA architecture as an implementation model for an Intelligent Environment has been described in Chapter 3. Further studies that could arise from this work include the modelling of sharing context rules between Intelligent Environment domains in the distributed knowledge-based context and the requirement to have flexible and efficient communication between resources managers in several Intelligent Environment domains in the formation of a society of Intelligent Environment domains.

Location awareness in an Intelligent Environment including the users' locations in an Active Office, which can be placed in three categories, i.e. precise location, proximate location, predicted location, has been described in Chapter 4. The categories are based on sensor capability to sense the area and the use of history data. Further studies that can be considered as arising from this work include:

a. Developing IP multicast system with group-content-filtering for data communication between Intelligent Environments. The distributed content-routing and filtering system (such as Elvin or Spread) can be used for an automatic update of object content in Intelligent Environment repository, in particular for updating user location.
b. Integration with desirable features of context-aware system speech input and output.

The instance-based learning method in the machine learning area can be used to classify and then estimate symbolic user location in the Ubiquitous Environment. The *ηk*-Nearest Neighbour algorithm, which is an instance-based learning algorithm, normalises the sample data set of the WiFi signal strength and signal quality to estimate symbolic user location at a room scale and has been described in Chapter 5. This chapter also proposed the use of Boolean MaxMin algorithm to determine the quality of the training data-set and discusses how to find the maximum common value to achieve the best correct result in the estimation user location.

Further studies which may arise from this work include:

1. This work uses off-line learning based on the learning data set. It can be developed into online learning by using extracts of the feature/function from the instance online signal data.
2. The symbolic user location, as the final output of this work, has been used for an Active Office in satisfaction of low level accuracy. Further study can develop an algorithm to achieve higher level accuracy and precision in estimating *coordinate user locations.* The Gaussian process models, Principal Component Analysis

(PCA) or the multivariate regression for signal strength may be explored for this purpose.
3. Machine learning in this work uses the Euclidian distance (standard distance), other distance methods can be used to increase the correctness of the estimation of the symbolic user location, such as Manhattan distance, Canberra distance, quadratic distance, and other relevant distances.

The user mobility model developed a user mobility pattern by using the history of user location and detecting the user's changing locations. The user mobility pattern can be formed by the number of visits and time spent in a one day observation and also by the use of mobility's direct graph. The *regularity* of user movement can be studied from the user mobility patterns for the prediction of future user movement. The user mobility pattern improved the accuracy continuously by *pattern of accuracy*, i.e., increasing the degree of accuracy of user mobility by comparing the regularity of the user mobility pattern and the current user mobility, as described in Chapter 6. Further studies to be considered from this work are:

- The deeper study of the *regularity* of user mobility by the use of probability or fuzzy logic theory.
- Study of the *pattern of accuracy* of user mobility. The pattern of user mobility can be studied more deeply to determine the sensitivity of the accuracy necessary for changes to regularity levels to increase the level of regularity.

The framework necessary for an Intelligent Environment to respond intelligently to user activities in a context-awareness environment required the design of a fusion sensor database and the modelling of the social environment to respond to user activities and were also presented in Chapter 8. Further studies to be considered as flowing from this work by adding richer context information to an Active Office will enable more responses to user activities, thus making even more 'intelligent' response behaviour possible.

Moreover, the response of the Intelligent Environment to the user in a social context model can be improved by developing the adaptation capability of the intelligent response. The environment response adaptation capability can be created by the use of machine learning, not only for the estimation of user location but also to enable the environment to have adaptation capability in responding to user activity. For example, when a user changes his room in the same office, the response from the environment to the user also follows him to the new room. In other words, changes to Active Office behaviour/response that follow user moves are on the user pattern and not based on a room.

10.4 Conclusions

An Intelligent Environment requires a context-aware application to work better and increase productivity and enjoyment for users as a result. The context awareness mechanism has four fundamental requirements i.e. identity (who), activity (what), location (where) and timestamp (when). This thesis proposed DiCPA architecture to manage scalable distribution of context processing in the Intelligent Environment. This architecture gives a complete set framework in developing an Intelligent Environment. In this study, the implementation model of an Intelligent Environment, which is an Active

Office, has not been developing all capability of the framework yet. However, based on this framework, the model of user location, user mobility, user activity and Intelligent Environment response were developed. Some prototypes were also developed to proof of Context-Aware Computing concept in the Intelligent Environment.

The fundamental concept of Context-Aware Computing's capacity to form user context for an Intelligent Environment required the physical *presence* of a user and three fundamental predicate relations, as proposed in Chapter 8, as follows:

- *Location Awareness* which leads to *User Mobility*
- *Activity Awareness* which leads to *User Situation*
- *Response Awareness* which leads to *Action of User Situation* from the environment

The evaluation strategy for research in the Intelligent Environment required interesting *core content* and provided *evaluation* processes. The evaluation process has iterative evaluation of the design process and of the product. The *software product quality metrics for Context-Aware Computing*, based on Boehm Model, McCall Model, new interpretation of ISO/IEC 9126 and some additional of metrics, are also proposed. The building of product quality metrics is influenced by two aspects: the technological and the social aspects, and two dimensions: the user and environment dimensions. The evaluation strategy for Context-Aware Computing above can be used as a standard evaluation for better understanding the performance of good quality Context-Aware Computing research in objective and meaningful measurement.

Bibliography

The bibliography in this study has two parts: Part 1. Cited Bibliography (References) and Part 2. Uncited Bibliography, which is useful as background reading for parts of this study.

1. Cited Bibliography (References)

Abowd, G. D. (1999). "Classroom 2000: An Experiment with the Instrumentation of a Living Educational Environment." IBM Systems Journal 38(4): pp. 508-530.

Abowd, G. (1999). "Software Engineering Issues for Ubiquitous Computing". In proceedings of International Conference on Software Engineering (ICSE'99), Los Angeles, California, United States. pp. 75 - 84.

Abowd, G. D., C. G. Atkeson, A. Feinstein, C. Hmele, R. Kooper, S. Long, N. Sawhney, M. Tani (1996). "Teach and Learning a Multimedia Authoring: The Classroom 2000 Project." Proceedings of the ACM Multimedia'96 Conference, Boston, MA. pp. 187-198.

Abowd, G. D., A. K. Dey, P. J. Brown, N. Davies, M. Smith, P. Steggleset (1999). "Towards a Better Understanding of Context and Context-Awareness." Proceedings of The 1st international symposium on Handheld and Ubiquitous Computing, Karlsruhe, Germany, Springer-Verlag. pp. 304-307.

Abraham, T. and J. F. Roddick. (1999). "Survey of Spatio-Temporal Databases." GeoInformatica 3(1): pp. 61-99.

Akyildiz, I. F. and W. Wang (2004). "The Predictive User Mobility Profile Framework for Wireless Multimedia Networks." IEEE Transactions on Networking 12(6): pp. 1021-1035.

Alam, L. S., D. Walker, P. Collings (2004). "Ubiquitous Computing Support for Student Group Work." The 15th Australasian Conference on Information Systems, Hobart, Tasmania. pp. 3.

AT&T-Lab-Cambridge (2001). Sentient Computing Project. Cambridge, UK. available at http://www.uk.research.att.com/spirit/.

Bagrodia, R., W. W. Chu, L. Kleinrock, G. Popek (1995). "Vision, Issues, and Architecture for Nomadic Computing." IEEE Personal Communications 2(6): pp. 14-26.

Bahl, P. and V. N. Padmanabhan (2000). "Radar: An in-building RF-based user location and tracking system." Proceedings of the IEEE Infocom 2000, Tel-Aviv, Israel Vol. 2. pp. 775-784.

Bellavista, P., A. Corradi, C. Stefanelli (2000). "A Mobile Agent Infrastructure for Terminal, User, Resource Mobility." In Proceedings of The Seventh IEEE/IFIP Network Operations and Management Symposium (NOMS), Honolulu, HI, IEEE Press, Piscataway, NJ. pp. 877-890.

Benerecetti, M., O. P. Bouquet, C. Ghidini (2000). "Contextual Reasoning Distilled." Journal of Experimental and Theoretical Artificial Intelligence (JETAI) 12(2000): pp. 279-305.

Berchtold, S., B. Ertl, D. A. Keim, H.P. Kriegel, T. Seidl (1998). “Fast Nearest Neighbor Search in High-Dimensional Space.” 14th International Conference on Data Engineering (ICDE 1998), Orlando, Florida, USA. pp. 209-219.

Berson, A. (1996). *Client/Server Architecture.* Second edition, McGraw-Hill Series on Computer Communications. 596 pages.

Bevan, N., J. Kirakowskib, J. Maissela (1991). “What is Usability?” Proceedings of the 4th International Conference on HCI, Stuttgart, Germany.

Boehm, B. W., J. R. Brown, M. Lipow (1976). “Quantitative Evaluation of Software Quality.” Proceedings of the 2nd International Conference on Software Engineering, San Francisco, California, USA, IEEE Computer Society Press. pp. 592-605.

Brooks, R. A. (1997). “The Intelligent Room Project.” Proceedings of the second International Cognitive Technology Conference (CT'97), Aizu, Japan. pp. 271-278.

Brown, P. J., J. D. Bovey, (1997). “Context Aware applications: From the laboratory to the marketplace.” IEEE Personal Communication 4(5): pp. 58-64.

Brumit, B. L., B. Meyer, X. Chen (2000). “EasyLiving: Technologies for Intelligent Environments.” 2nd International Symposium on Handheld and Ubiquitous Computing (HUC 2000), Bristol, UK. pp. 12-27.

Budzik, J. and K. J. Hammod (2000). “User Interactions with Everyday Applications as Context for Just-in-time Information Access.” Proceedings of the International Conference on Intelligent User Interfaces, New Orleans, Louisiana, USA. pp. 44-51.

Carroll, J. J., D. C. Neale, P. L. Isenhour, M. B. Rosson, D. S. McCrickard (2003). “Notification and awareness: synchronizing task-oriented collaborative activity.” International Journal of Human-Computer Studies 58: pp. 605-632.

Castro, P., P. Chiu, T. Kremenek, R. Muntz (2001). “A Probabilistic Room Location Service for Wireless Networked Environments.” Third International Conference on Ubiquitous Computing (UbiComp 2001), Atlanta, Georgia. pp. 18-34.

Cavano, J. P. and J. A. McCall (1978). “A Framework for the Measurement of Software Quality.” ACM SIGSOFT Software Engineering Notes, ACM Press Vol. 3 (5). pp. 133-139.

Cayirci, E. and I. F. Akyildiz (2002). “User Mobility pattern Scheme for Location Update and Paging in Wireless Systems.” IEEE Transactions on Mobile Computing 1(3): pp. 236-247.

Chan, A. T. S., S. N. Chuang, J. Cao, H. V. Leong (2004). “An Event-Driven Middleware for Mobile Context Awareness.” The Computer Journal 47(3): pp. 278-288.

Chan, J. and A. Seneviratne (1999). “A Practical User Mobility Prediction Algorithm for Supporting Adaptive QoS in Wireless Network.” The IEEE International Conference on Networks (ICON'99). pp. 104-111.

Cheverst, K., N. Davies, K. Mitchell, A. Friday, C. Efstratiou (2000). “Developing a Context-aware Electronic Tourist Guide: Some Issues and Experiences.” Proceedings of the 6th annual international conference on Mobile computing and networking, Boston, Massachusetts, United States. pp. 20 - 31.

Chuang, S. N., A. T. S. Chan, J. Cao (2002). “Dynamic Service Composition for Wireless Web Access.” International Conference on Parallel Processing, Vancouver, British Columbia, Canada. pp. 429-436.

Churchill, E. F. and A. J. Munro (2001). “WORK/PLACE: Mobile Technologies and Arenas of Activities.” SIGGROUP Bullettin Vol. 22(3): pp. 3-9.

Cisco-Inc. (2002). “iHome: Australia's first internet-enabled home”. 15 May 2002. Available at http://www.iHome.com.au.

Coen, M. H. (1998). “Design Principles for Intelligent Environments.” Proceedings of the 1998 National Conference on Artificial Intelligent (AAAI-98), Madison, Wisconsin, USA. pp. 547-554.

Coen, M. H. (1999). “The Future of Human-Computer Interaction, or how I learn to stop worrying and love my Intelligent Room” IEEE Intelligent Systems 14(2): pp. 8-10.

Covington, M. J., W. Long, S. Srinivasan, A.K. Dey, M. Ahamad, G.D. Abowd, (2001). “Securing Context-Aware Applications Using Environment Roles.” Proceedings of The Sixth ACM Symposium on Access Control Models and Technologies (SACMAT'01), Chantilly, Virginia, USA. pp. 10-20.

Crabtree, A., T. Rodden, T. Hemmings, S. Benford (2003). “Finding a Place for Ubicomp in the Home.” Proceedings of The fifth International Conference on Ubiquitous Computing (Ubicomp'03), LCNS 2864, Seattle, USA, Springer Verlag. pp. 208-226.

Crowley, J. L. (2003). “Context Driven Observation of Human Activity.” The first European Symposium on Ambient Intelligence (EUSAI) 2003, LNCS 2875, Springer-Verlag Berlin Heidelberg. pp. 101-118.

Cui, Y., K. Nahrstedt, D. Xu (2004). “Seamless User-Level Handoff in Ubiquitous Multimedia Service Delivery.” Multimedia Tools and Applications 22: pp. 137-170.

Daniel, R. and M. Mealling (1997). “Resolution of Uniform Resource Identifiers using the Domain Name System.” RFC 2168.

Dar, S., M. J. Franklin, B. T. Johnson, D. Srivastava, M. Tan (1996). “Semantic Data Caching and Replacement.” Proceedings of the 22nd International Conference on Very Large Data Bases (VLDB), Mumbai (Bombay), Morgan Kauffmann. pp. 330-341.

Date, C. J. (1995). *An Introduction to Database Systems.* Sixth edition, Addison Wesley. 296 pages.

Dazeley, R. and B. H. Kang (2004). “An Augmentation Hybrid System for Document Classification and Rating.” Smart Internet Technology Cooperative Research Centre, Report No. P03-026: 11 pages.

Del-Re, E., R. Fantacci, G. Giambene (2000). "Characterization of User Mobility in Low Earth Orbit Mobile Satellite Systems." Wireless Networks 6: pp. 165-179.

Demirdjian, D., K. Tollmar, K. Koile, N. Checka, T. Darrell (2002). “Activity maps for location-aware computing.” Sixth IEEE Workshop on Applications of Computer Vision, Orlando, Florida. Pp.70-75.

Dey, A. K. (2002). “Evaluation of Ubicomp Applications and Systems.” Summer School on Ubiquitous and Pervasive Computing. Schloss Dagstuhl, Germany.

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

Dey, A. K., G. D. Abowd, D. Salber (1999). “A Context-Based Infrastructure for Smart Environments.” Proceedings of the First International Workshop on Managing Interaction in Smart Environments (MANSE'99), Dublin, Ireland. pp. 114-128.

Downtown, A. (1991). *“Engineering The Human Computer Interface.”* Essex, UK, McGraw Hill. pp. 269-270, 334.

Dyreson, C. E., R. T. Snodgrass, M. Freiman (1995). “Efficiently supporting temporal granularities in a DBMS.” Technical Report TR 95/07, James Cook University: 22 pages.

Ducatel K., M. Bogdanowicz, F. Scapolo, J. Leijten, J. C. Burgelman (2001). “ISTAG report: Scenario for Ambient Intelligence in 2010.” European Commission Community Centre. Available at http://www.cordis.lu/ist/istag.htm.

ECIS-Directorate-General (2001). “ISTAG report: Scenario for Ambient Intelligence in 2010.” ECIS-Directorate-General. Available at http://www.cordis.lu/ist/istag.htm.

Ekahau (2005). Ekahau Inc. Avalable at http://www.ekahau.com/.

Elrod, S., G. Hall, R. Constanza, M. Dixon, J. D. Riviered (1993). “Response Office Environments.” Communications of the ACM 36(7): pp. 84-85.

Engestrom, Y. (1987). “Learning by Expanding: An Activity Theory Approach to Developmental Research.” Helsinki, Orienta-Konsultit. pp. 368.

Fithian, R., G. Iachello, J. Moghazy, Z. Pousman, J. Stasko (2003). “The Design and Evaluation of a Mobile Location-Aware Handheld Event Planner.” in Proceedings of the 5th International Symposium on Human-Computer Interaction with Mobile Devices and Services, Mobile HCI 2003, Udine, Italy, LNCS 2795, Springer Verlag. pp. 145-160.

Fitzpatrick, A., G. Biegel, S. Clarke, V. Cahill (2002). “Toward a Sentient Object Model.” Workshop on Engineering Context-Aware Object-Oriented Systems and Environments (ECOOSE), Seattle, WA, USA.

Flanagan, J. L. (1999). “Autodirective Sound Capture: Towards Smarter Conference Rooms.” IEEE Intelligent Systems 14(2): pp. 16-19.

Flickenger, R. (2002). *Building Wireless Community Networks.* O'Reilly & Associates, Inc. pp. 8-15.

Frees, S. E. and G. D. Kesser (2003). “Characterizing User Mobility within an Immersive Virtual Environment.” Bethlehem, PA, USA, Department of Computer Science and Engineering, Lehigh University.

Garmin. (2005). Garmin® GPS, Garmin Inc. Available at http://www.garmin.com/about GPS/.

Gray, W. D. and M. C. Salzman (1998). “Damaged Merchandise? A Review of Experiment that Compare Usability Evaluation Methods.” Human-Computer Interaction 13: pp. 203-262.

Gutwin, C. and S. Greenberg (1996). “Workspaces Awareness for Groupware.” The proceeding of the ACM HCI'96 Conference on Human Factors in Computing Systems, New York, Association of Computing Machinery. pp. 208-209.

Hallberg, J. and M. Nilsson (2002). *Positioning with Bluetooth, IrDa and RFID.* Masters Thesis. Department of Computer Science and Electrical Engineering, Lulea University of Technology: 58 pages.

Harle, R. and G. Coulouris (2002). "A Grand Challenge in Computer Science: The Sentient Building (Part I)." In The Workshop on Grand Challenges for Computing Research, e-Science Institute, Edinburgh, Scotland.

Harren, M., J. M. Hellenstein, R. Huebsch (2002). "Complex Queries in DHT-based Peer-to-Peer Networks." International Workshop on Peer-to-Peer System (IPTPS'02), Cambridge, USA. pp. 242-250.

Harter, A. and A. Hopper (1994). "A Distributed Location System for the Active Office." IEEE Network 8(1): pp. 62-70.

Harter, A., A. Hopper, P. Steggles, A. Ward, P. Webster (2001). "The Anatomy of a Context-Aware Application." Wireless Networks 1: pp. 1-16.

Hartson, H. R., T. S. Andre, R. C. Williges (2003). "Criteria for Evaluating Usability Evaluation Methods." International Journal of Human-Computer Interaction 15(1): pp. 145-181.

Hightower, J. and G. Borriello (2004). "Particle Filter for Location Estimation in Ubiquitous Computing: A Case Study." Proceedings of The Sixth International Conference on Ubiquitous Computing (UbiComp 2004), LNCS 3205, Nottingham, UK, Springer Verlag. pp. 88-106.

Hong, J. I. and J. A. Landay (2001). "An Infrastructure Approach to Context-Aware Computing." Human-Computer Interaction (HCI) Journal 16(2,3 & 4). pp. 287-303.

Hopper, A. (1999). "Sentient Computing." The Royal Society Clifford Paterson Lecture, Philosophical Transactions, Royal Society London, Volume 358: pp. 1349-2358.

Hull, R., P. Neaves, J, Bedford-Roberts (1997). "Towards Situated Computing." Proceedings of The 1st International Symposium on Wearable Computers (ISWC'97), Cambridge, Massachusetts, IEEE Computer Society Press. pp. 146-153.

Indulska, J. and P. Sutton (2003). "Location Management in Pervasive Systems." Workshop on Wearable, Invisible, Context-Aware, Ambient, Pervasive and Ubiquitous Computing, ACSW 2003, Adelaide, Australia, Australian Computer Science Communications Vol. 25 (6). pp. 143-152.

Ipina, D. L. D. (2000). "Building Components for a Distributed Sentient Framework with Python and CORBA." Online Proceedings of the 8th International Python Conference, Arlington, VA, USA. Available at http://phyton.fyxm.net/ workshops/2000-01/proceedings.html

Ipina, D. L. D. (2001). "An ECA Rule-Matching Service for Simpler Development of Reactive Applications." Supplement of the Proceedings of Middleware 2001, IEEE Distributed Systems Online Vol. 2 (7).

Ipina, D. L. D. and S. L. Lo (2001). "Sentient Computing for Everyone." The Third IFIP WG 6.1 International Working Conference on Distributed Applications and Interoperable Systems (DAIS 2001), Krakow, Poland. pp. 41-54.

ISO/IEC-9126 (1991). "Software Product Evaluation - Quality Characteristics and Guidelines for Their Use." ISO DIS 9126.

Jensen, C. S. (2000). *Temporal Database Management.* dr. techn Thesis, Aalborg University, Denmark: 1328 pages.

Jensen, C. S., J. Kolarvr, T. B. Pedersen, I. Timko (2003). “Nearest Neighbor Queries in Road Networks.” Proceedings of the Eleventh ACM International Symposium on Advances in Geographic Information Systems (ACMGIS 2003), New Orleans, Louisiana, USA. pp. 1-8.

Jiang, C. and P. Steenkiste (2002). “A Hybrid Location Model with a Computable Location Identifier for Ubiquitous Computing.” Proceedings of the Fourth International Conference on Ubiquitous Computing (Ubicomp 2002), Goteborg, Sweden. pp. 246-263.

Johnson, C. W., D. Carmichael, J. Kay, B. Kummerfeld, R. Hexel (2004). “Context Evidence and Location Authority: The Disciplined Management of Sensor Data Into Context Models.” First International Workshop on Advanced Context Modelling, Reasoning and Management, the Sixth International Conference on Ubiquitous Computing (UbiComp 2004), Nottingham, UK.

Johanson, B., A. Fox, 2001. “Tuplespaces as Coordination Infrastructure for Interactive Workspaces”. In: UbiTools'01–Workshop on Application Models and Programming Tools for Ubiquitous Computing held in conjunction with the Third International Conference on Ubiquitous Computing (Ubicomp 2001).

Johanson, B., A. Fox, T. Winograd, (2002). “The Interactive Workspaces Project: Experiences with Ubiquitous Computing Rooms”, IEEE Pervasive Computing, special issue on "Integrated Pervasive Computing Environments" 1 (2), 67–74.

Juan, T., A. Pearce, L. Sterling (2002). “ROADMAP: Extending the Gaia Methodology for Complex Open Systems of BDI agents.” Proceedings of the third International Workshop on Agent-Oriented Software Engineering, at Autonomous Agents & Multi-Agent Systems (AAMAS) 2002, Bologna, Italy. pp. 3-10.

Juan, T., L. Sterling, M. Winikoff (2002). “Assembling Agent Oriented Software Engineering Methodologies from Features.” Proceedings of the third International Workshop on Agent Oriented Software Engineering (AOSE), at Autonomous Agents & Multi-Agent Systems (AAMAS) 2002, Bologna, Italy. pp. 198-209.

Käppeli, D. (2003). *JXTA over Bluetooth.* Diploma Thesis, Eidgenössische Technische Hochschule Zürich: 81 pages.

Kern, N., B. Schiele, A. Schmidt (2003). “Multi-sensor Activity Context Detection for Wearable Computing.” The first European Symposium on Ambient Intelligence (EUSAI), Springer-Verlag Berlin Heidelberg, LNCS 2875. pp. 220-232.

Kidd, C. D., R. Orr, G. D. Abowd, C. G. Atkeson, I.A. Essa, B. MacIntyre, E. Mynatt, T.E. Starner, W. Newstetter (1999). “The Aware Home: A living Laboratory for Ubiquitous Computing Research.” Proceedings of The 2nd International Workshop on Cooperative Building (CoBuild'99), LNCS 1670, Pittsburgh, PA, Springer Verlag. pp. 191-198.

Kleinrock, L. (1995). “Nomadic Computing - An Opportunity.” ACM SIGCOMM Computer Communication Review 25(1): pp. 36-40.

Kleinrock, L. (2000). “On Some Principles of Nomadic Computing and Multi-Access Communications.” IEEE Communications Magazine: pp. 46-50.

Kleinrock, L. (2001). "Breaking Loose." Communications of the ACM 44(9): pp. 41-45.

Knuth, D. E. (1997). *The Art of Computer Programming: Fundamental Algorithms*. Third ed. Redwood City, CA-USA, Addison Wesley Longman Publishing Co., Inc.

Kohonen, T. (1995). *The Self-Organizing Map*. Berlin, Springer.

Koile, K., K. Toolman, D. Demirdjian, H. Shrobe, T. Darrell (2003). "Activity Zones for Context-Aware Computing." Proceedings of The 5th International Conference on Ubiquitous Computing (Ubicomp'03), LNCS 2864, Seatle, USA, Springer-Verlag. pp. 90-103.

Kolahdouzan, M. and C. Shahabi (2004). "Voronoi-Based K Nearest Neighbor Search for Spatial Network Databases." Proceedings of the Thirtieth International Conference on Very Large Data Bases (VLDB 2004), Toronto, Canada, Morgan Kaufmann. pp. 840-851.

Korn, F., N. Sidiropoulos, C. Faloutsos, E. Siegel, Z. Protopapas (1996). "Fast Nearest Neighbor Search in Medical Image Databases." Proceedings of The 22th International Conference on Very Large Data Bases (VLDB 1996), Mumbai (Bombay), India., Morgan Kaufmann. pp. 215-226.

Kravets, R., C. Carter, L. Magalhaes (2001). "A Cooperative Approach to User Mobility." ACM Computer Communications Review 31(5): pp. 57-69.

Kulkarni, A. (2002). *A Reactive Behavioural System for the Intelligent Room.* Masters Thesis. Computer Science. Cambridge, MA, Massachusetts Institute of Technology: 63 pages.

Kummerfeld, B., A. Quigley, C. W. Johnson, R. Hexel (2003). "Towards an Intelligent Environment Architecture for Multi-granularity Context Description." Workshop on User Modelling for Ubiquitous Computing, Pittsburgh, PA, USA.

LCE (2002). "Sentient Computing." Cambridge, UK, LCE-Laboratory for Communications Engineering-Cambridge.

Lee, S. W. and K. Mase (2002). "Activity and Location Recognition using Wearable Sensors." IEEE Pervasive Computing (July-Sept 2002): pp. 24-31.

Leeuwen, J. V. (2002). "Approaches in Machine Learning." In *Algorithms in Ambient Intelligence* (Chapter 8). Edited by W.Verhaegh, E.Aarts and J.Korst. Dordrecht, Kluwer Academic Publishers. 2: pp. 155-166.

Lindwer, M., D. Marculescu, T. Basten, R. Zimmermann, R. Marculescu, S. Jung, E. Cantatore (2003). "Ambient Intelligence Vision and Achievements: Linking Abstract Ideas to Real-World Concepts." Proceedings of The Design, Automation and Test in Europe Conference and Exhibition (DATE'03), Munich, Germany. pp. 10-17.

Liu, T., P. Bahl, I. Chlamtac (1998). "Mobility Modeling, Location Tracking, and Trajectory Prediction in Wireless ATM Networks." IEEE on Selected Areas In Communications 16(6): pp. 922-936.

Loenen, E. J. V. (2003). "On the Role of Graspable Objects in the Ambient Intelligence Paradigm." Proceedings of Smart Object Conference (SOC'03), Grenoble, France.

Lukowicz, P., J. A. Ward, H. Junker, M. Stager, G. Troster, A. Atrash, T. Starner (2004). "Recognizing Workshop Activity Using Body Worn Microphones and Accelerometers." Pervasive 2004, Springer-Verlag Berlin Heidelberg, LNCS 3001. pp. 18-32.

Lund, A. M. (1998). “Damaged Merchandise? Comments on shopping at outlet malls.” Human-Computer Interaction 13: pp. 276-281.

Mantoro, T. (1994). *Self-Organizing Map Approach for the Discovery of Shared Interest.* Masters Thesis. Department of Computer Science. Bangkok, Thailand, Asian Institute of Technology: 96 pages.

Mantoro, T., and C. W. Johnson (2003). “User Mobility Model in an Active Office.” LNCS 2875, The first European Symposium on Ambient Intelligence (EUSAI’03), Eindhoven, The Netherlands. pp. 42-55.

Mantoro, T. and C. W. Johnson (2003). “Location History in a Low-cost Context Awareness Environment.” Workshop on Wearable, Invisible, Context-Aware, Ambient, Pervasive and Ubiquitous Computing, ACSW 2003, Adelaide, Australia, Australian Computer Science Communications Vol. 25 (6). pp. 153-158.

Mantoro, T. and C. W. Johnson (2003). “User Location and Mobility for Distributed Intelligent Environment.” Adjunct Proceedings, The Fifth International Conference on Ubiquitous Computing (UbiComp’03), Seattle, Washington, USA. pp. 12-15 October 2003.

Mantoro, T. and C. W. Johnson (2004). “DiCPA: Distributed Context Processing Architecture for an Intelligent Environment.” The Communication Networks and Distributed Systems Modelling Conference (CNDS’04), San Diego, California.

Mealling, M. and R. Daniel (2000). “The Naming Authority Pointer (NAPTR) DNS Record.” Updated RFC 2168.

Mitchell, T. M. (1997). *Machine learning.* Boston, MA, WCB/McGraw-Hill.

Mozer, M. C. (1999). “An Intelligent Environment Must be Adaptive.” IEEE Intelligent Systems 14(2): pp. 11-13.

Nielsen, J. (1994). “Guerrilla HCI: Using Discount Usability Engineering to Penetrate the Intimidation Barier.” In *Cost-Justifying Usability.* Edited by R. G. Bias and D. J. Mayhew.

Nielsen, J. and J. Levy (1994). “Measuring Usability: Preference vs. Performance.” Communications of the ACM 37(4): pp. 67-76.

Nieuwoudt, C. and E. C. Botha (2002). "Cross-language use of acoustic information for automatic speech recognition." Speech Communication 38: pp. 101-113.

Olson, G. M. and T. P. Moran (1998). “Commentary on "Damaged Merchandise?".” Human-Computer Interaction 13: pp. 263-323.

Orr, R. J. and G. D. Abowd (2000). “The Smart Floor: A mechanism for Natural User Identification and Tracking.” Proceedings of the Conference on Human Factors in Computing Systems (CHI '00), The Hague, Netherlands, ACM Press. pp. 275-276.

Padgham, L. and M. Winikoff (2002). “Prometheus: A Methodology for Developing Intelligent Agents.” Proceedings of the Third International Workshop on Agent-Oriented Software Engineering, at AAMAS 2002, Bologna, Italy, ACM. pp. 37-38.

Papadias, D., J. Zhang, N. Mamoulis, Y. Tao (2003). “Query Processing in Spatial Network Databases.” Proceedings of 29th International Conference on Very

Large Data Bases (VLDB 2003), Berlin, Germany, Morgan Kaufmann. pp. 802-813.

Parent, C., S. Esteban, Z. Spaccapietra (1999). “Spatio-Temporal Conceptual Models: Data Structures + Space + Time.” 7th ACM International Symposium on Advances in Geographic Information Systems, Kansas City, USA.

Pascoe, J. (1998). “Adding Generic Contextual capabilites to Wearable Computer.” Proceedings of The 2nd International Symposium on Wearable Computers (ISWC 1998), Pittsburgh, Pennsylvania, USA, IEEE Computer Society. pp. 92-99.

Patterson, D. J., D. Fox, H. Kautz, M. Philipose (2003). “Expressive, Tractable and Scalable Technique for Modelling Activities of Daily Living.” UbiHealth 2003: The 2nd International Workshop on Ubiquitous Computing for Pervasive Healthcare Applications, Seattle, WA.

Picard, R. (1995) “Affective computing.” Technical Report 321, MIT Media Lab, Perceptual Computing. Available at http://vismod.www.media.mit.edu/vismod.

Ponnekanti, S., B. Johanson, E. Kiciman, and A. Fox (2003) A. Portability, Extensibility and Robustness in iROS. In Proceeding IEEE International Conference on Pervasive Computing and Communications (PerCom 2003). Pp.11-19.

Porkaew, K., I. Lazaridis, S. Mehrotra (2001). “Querying Mobile Objects in Spatio-Temporal Databases.” International Symposium on Spatial and Temporal Databases (SSTD) 2001. LNCS 2121. pp. 59-78.

Preece, J., Y. Rogers, H. Sharp, D. Benyon, S. Holland, and T. Carey (1994). *Human-Computer Interaction*. Addison-Wesley. pp. 401-402, 517, 722-723.

Prekop, P. and M. Burnett (2002). “Activities, Context and Ubiquitous Computing.” Computer Communications 26(11): pp. 1168-1176.

Pressman, R. S. (2005). *Software Engineering: A Practitioner's Approach.* 6th edition. Singapore, McGraw-Hill. pp. Pages: 880.

Priyantha, N. B., A. Chakraborty, H Balakrishnan (2000). “The Cricket Location-Support System.” Proceeding of The 6th ACM international Conference on Mobile Computing and Networking (MOBICOM 2000), Boston, MA, ACM. pp. 32-43.

Rebman, C. M. J, M. W. Alken, C. G. Cegielski (2002). "Speech Recognition in the Human-computer Interface." Information and Management 40(6): pp. 509-519.

Ren, Q. and M. H. Dunham (2000). “Using Semantic Caching to Manage Location Dependent Data in Mobile Computing.” Proceedings of The 6th International Conference Mobile Computing and Networking (MOBICOM 2000), New York, ACM Press. pp. 210-221.

Roussopoulos, N., S. Kelly, F. Vincent (1995). “Nearest Neighbor Queries.” Proceedings of The 1995 ACM SIGMOD International Conference on Management of Data, San Jose, California, USA, ACM Press. pp. 71-79.

Sandhu, R. S., E. J. Coyne, H. L. Feinstein, C. E. Youma (1996). “Role-Based Access Control Models.” IEEE Computer 29(2): pp. 38-47.

Satchell, C. and S. Singh (2004). “User Problems - Design Solutions Swarms for Nomads.” Workshop on Ubiquitous Systems for Suporting Social Interaction and Face-to-Face Communication in Public Spaces, at the Sixth International Conference on Ubiquitous Computing (Ubicomp 04), Nottingham, UK. pp. 29-32.

Satyanarayanan, M. (2001). “Pervasive Computing: Vision and Challenges.” IEEE Personal Communications 8(4): pp. 10-17.

Schilit, B. and M. Theimer (1994). “Disseminating Active Map Information to Mobile hosts.” IEEE Network 8(5): pp. 22-32.

Schilit, B., N., Adams, R. Want (1994). “Context-Aware Computing Applications.” Proceedings of IEEE Workshop on Mobile Computing Systems and Applications. Santa Cruz, CA: pp 85-90.

Schilit, W. N. (1995). *A System Architecture for Context-Aware Mobile Computing.* PhD Thesis. The Graduate School of Arts and Sciences. Colombia, Colombia University: 144 pages.

Schmidt, A. (2002). *Ubiquitous Computing - Computing in Context.* Ph.D Thesis. Computing Department. Lancaster, U.K., Lancaster University: pp. 294.

Scholtz, J. (2001). “Evaluation Methodologies for Context-Aware Computing.” The CHI 2001 workshop on Distributed and Disappearing User Interfaces in Ubiquitous Computing at UbiComp'01, Seattle, Washington, USA.

Scriven, M. (1967). “The Methodology of Evaluation.” In R. Tyler, R. Gagne and Scriven M. (Eds.). *Perspective of Curriculum Evaluation*. Chicago, USA, Rand Mc Nally. pp. 39-83.

Segall, B., D. Arnold, J. Boot, M. Henderson, T. Phelps (2000). “Content Based Routing with Elvin4.” Proceedings of the Australian Unix User Groups 2000 (AUUG2K), Canberra, Australia.

Seidl, T. and H. P. Kriegel (1998). “Optimal Multi-Step k-Nearest Neighbor Search.” Proceedings of the 1998 ACM SIGMOD International Conference on Management of Data, Seattle, Washington, USA. ACM Press. pp. 154-165.

Shafer, S. A. N. (2003). “Location Authorities for Ubiquitous Computing.” Proc. 2003 Workshop on Location-Aware Computing (part of Ubicomp 03), Seattle, Washington. pp. 13-15.

Shen, X., J. W. Mark, j. Ye (2000). “User Mobility Profile Prediction: An Adaptive Fuzzy Inference Approach.” Wireless Networks 6: pp. 363-374.

Smailagic, A. and D. Kogan (2002). "Location Sensing and Privacy in a Context-Aware Computing Environment." IEEE Wireless Communications 9(5): pp. 10-17.

Snyder, L., F. Baskett, M. M. Carroll, D. M. Coleman, D. Estrin, M. L. Furst, J. Hennessy, H. T. Kung, K. Maly, B. Reid (1994). *Academic Careers for Experimental Computer Scientists and Engineers.* Washington D.C., National Academy Press. 139 pages

Sousa, J. P. and D. Garlan (2002). “Aura: An Architectural Framework for User Mobility in Ubiquitous Computing Environments.” Proceedings of the 3rd Working IEEE/IFIP Conference on Software Architecture, Kluwer Academic Publishers. pp. 29-43.

Spasojevic, M. and T. Kindberg (2001). “Evaluating the CoolTown User Experience.” Workshop on Evaluation Methodologies for Ubiquitous Computing at UbiComp'01, Seattle, Washington, USA.

Stern, E. (2003). “Action Research & Evaluation: Performance, Learning & Understanding?” Presentation to Cabinet Office Evaluation Seminar Series, The Tavistock Institute. http://www.policyhub.gov.uk/docs/Action%20Research%20-%20Eliott%20Stern.pdf.

Stinson, L. L. (1999). "Measuring How People Spend Their Time: a Time-Use Survey Design." Monthly Labor Review (August 1999): pp. 12-19.

Strang T., C. Linnhoff-Popien (2004) “A Context Modeling Survey.” Workshop on Advanced Context Modelling, Reasoning and Management as part of The Sixth International Conference on Ubiquitous Computing (UbiComp 2004), Nottingham/England.

Szalai, A. (1972). *The Use of Time: Daily Activities of Urban and Suburban Populations in Twelve Countries*. Paris, The Hague, Mouton.

Tabucanon, M. T. (1988). *Multiple Criteria Decision Making in Industry.* Amsterdam, The Netherlands, Elsevier.

Tang, T. J. (1991). “Finding from observational studies of collaborators.” International Journal of Man-Machine Studies 34: pp. 143-160.

Tapia, E. M., S. S. Intille, K. Larson (2004). “Activity Recognition in the Home Using Simple and Ubiquitous Sensors.” The 2rd International Conference on Pervasive Computing (Pervasive 2004), LNCS 3001, Berlin Heidelberg: Springer-Verlag. pp. 158-175.

Thomas, B. H., V. Demczuk, W. Piekarski, D. Hepworth, B. Gunther (1998). “A Wearable Computer System with Augmented Reality to Support Terrestrial Navigation.” Proceedings of The 2nd International Symposium on Wearable Computers (ISWC 1998), Pittsburgh, Pennsylvannia, USA, IEEE Computer Society. pp. 168-171.

Triantaphyllou, E. (2000). *Multi-Criteria Decision Making Methods: A Comparative Study*. Applied Optimization, Vol. 44. Springer, 320 pages.

Weiser, M. (1996). “Ubiquitous Computing.” Available at http://www.ubiq.com/hypertext/weiser/UbiHome.html. 17 April 2004.

Weiser, M. (1991). “The Computer for the 21st Century.” Scientific American vol. 265(No. 3): pp. 94-104.

Weiser, M. (1993). “Some Computer Science Issues in Ubiquitous Computing.” Communications of the ACM 6(7): pp. 75-84.

Weiser, M. (1998). “The Invisible Interface: Increasing the Power of the Environment through Calm Technology.” Opening Keynote Speech. Proceedings of The First International Workshop on Cooperative Buildings, Integrating Information, Organization, and Architecture (CoBuild'98), LNCS 1370, Darmstadt, Germany, Springer. p.1.

Winograd, T. (2001). “Architecture of Context.” Human-Computer Interaction 16: pp. 401-419.

Wooldridge, M., N. Jennings, D. Kiddy (2000). “The Gaia Methodology for Agent-Oriented Analysis and Design.” Journal of Autonomous Agents and Multi-Agent System 3(3): pp. 285-312.

Zonoozi, M. M. and P. Dassanayake (1997). “User Mobility Modelling and Characterization of Mobility Patterns.” IEEE Journal on Selected Areas in Communications 15(7): pp. 1239-1252.

2. Uncited Bibliography

Booch, G., J. Rumbaugh, I. Jacobson (2000). *The Unified Modeling Language User Guide.* 6th Edition. Reading, Massachusetts, Addison Wesley. 512 pages.

Bohlen, M. H. (1995). “Temporal Database System Implementations.” Proceedings of the 1995 ACM SIGMOD International Conference on Management of Data, SIGMOD Record 24(2), San Jose, California. pp. 53-60.

Burrell, J., G. K. Gay, K. Kubo, N. Farina (2002). “Context-Aware Computing: A Test Case.” Proceedings of the 4th international conference on Ubiquitous Computing, UbiComp'02. LNCS 2498, Göteborg, Sweden, Springer-Verlag London, UK. pp. 1 - 15.

Braude, E. J. (2001). *Software Engineering: An Object-Oriented Perspective.* Boston Univ., Wiley. 560 pages.

Cook, J. L. (2000). *WAP Servlets: Professional Developer's Guide.* Wiley. 416 pages.

Crowley, J. L. (2004). “Context Aware Observation of Human Activity.” In: Proceedings of SIPS 2004, Oulu, Finland. The Proceedings of SIPS 2004, Oulu, Finland.

Dey, A. K. (2000). *Providing Architectural Support for Building Context-Aware Applications.* PhD Thesis, Georgia Institute of Technology: 170 pages.

Dourish, P. (2001). “Seeking a Foundation for Context-Aware Computing.” Human-Computer Interaction 16(2,3 & 4): pp. 229-241.

DuBois, P. (2003). *MySQL.* 2nd Edition, Sams. 1248 pages.

English, C., P. McGettrick, H. Lowe (2002). Dynamic Trust Models for Ubiquitous Computing Environment. Workshop on Security in Ubiquitous Computing, UbiComp'02, Göteborg, Sweden. pp. 1-4.

Garlan, D. and B. Schmerl (2001). “Component-based Software Engineering in Pervasive Computing Environments.” In 4th Workshop on Component-Based Software Engineering: Component Certification and System Prediction (held during 23rd ICSE'01), Toronto, Canada.

Hopkins, B. and R. Antony (2003). *Bluetooth for Java.* Apress. 352 pages.

Jourdan, E. (1979). *Structured Design: Fundamentals of a Discipline of Computer Program and Systems Design.* Prentice Hall. 473 pages.

Knudsen, J. (2003). *Wireless Java: Developing with J2ME.* Second Edition, Apress. 384 pages.

Kumar, C. B., P. Kline, T. Thompson (2003). *Bluetooth Application Programming with the Java APIs.* Morgan Kaufmann. 498 pages.

Leonhardt, U. (1998). *Supporting Location-Awareness in Open Distributed Systems.* PhD and Diploma Thesis. Department of Computing. London, England, Univeirsity of London and Imperial College of Science, Technology and Medicine: 186 pages.

Milner, R. (2002). “Science for Global Ubiquitous Computing. Workshop on Grand Challenges for Computing Research.” Edinburg, Scotland. 14 pages.

Milner, R. (2002). “Theories for Ubiquitous Processes and Data: Platform for 15-year Grand Challenge.” Workshop on Grand Challenges for Computing Research, Edinburg, Scotland.

Oaks, S., B. Traversat, L. Gong (2002). JXTA in a Nutshell. 1st Edition. 416 pages.

Page-Jones, M. (1988). *The Practical Guide to Structured Systems Design.* Englewood Cliffs, New Jersey, Yourdon Press Computing Series, Prentice-Hall. 384 pages.

Philips, B. A. (1999). *Metaglue: A programming Language for Multi-Agent Systems.* Bachelor of Science and Masters Thesis. Department of Electrical Engineering and Computer Science. Massachusetts, US, Massachusetts Institute of Technology: 84 pages.

Ramage, M. (1999). *The Learning Way: Evaluating Co-operative Systems.* Ph.D. Thesis. Department of Computing. Lancaster, UK, Lancaster University: 142 pages.

Roman, G. C., G. P. Picco, A. L. Murphy (2000). "Software Engineering for Mobility: a Roadmap." International Conference on Software Engineering Proceedings of the Conference on The Future of Software Engineering. pp. 241 - 258.

Sharpe, R., T. Potter, J. Ye (2000). "Special Edition: Using Samba." 1st Edition, Que. 671 pages.

Siegel, J. (2000). *CORBA 3 Fundamentals and Programming.* 2nd Edition. New York, Wiley Computer Books. 928 pages.

Theriault, M., C. Claramunt, P. Y. Villeneuve (1999). "A Spatio-Temporal Taxonomy for the Representation of Spatial Set Behaviours." The International Conference on Spatio-Temporal Database Management (STDBM'99), LNCS 1679, Scotland, Edinburgh, Springer-Verlag Berlin Heidelberg. pp. 1-18.

Tuulari, E. (2000). "Context Aware Hand-held Devices." Publications 412, VTT Electronics Technical Research Centre of Finland: 83 pages.

Zambonelli, F. and H. V. D. Parunak (2003). "Signs of a Revolution in Computer Science and Software Engineering." In Proceedings of the 3rd International Workshop on Engineering Societies in the Agents World. LNCS 2577, New York, US, Springer-Verlag. pp. 13-28.

Index