A Context-Aware Framework for Personalised Recommendation in Mobile Environments

Kam Fung Yeung

A thesis submitted in partial fulfilment of the requirements for the award of the degree of

Doctor of Philosophy

University of Portsmouth

September 2011
Abstract

Context-awareness has become an essential part in various personalised applications such as mobile recommender systems and mobile information retrieval. Much progress has been made in context-aware applications. However there is a lack of general framework for supporting the rapid development of context-aware applications and enabling the sharing and dissemination of context information across different applications.

This dissertation presents a novel context-aware framework for supporting context distributions and personalised services in mobile environments. This dissertation makes four major contributions: First, it proposes a JXTA-based Hybrid Peer-to-peer framework, called JHPeer, for efficient organisation, representation, retrieval and management of context data, which enables rapid development of context-aware applications for mobile users. JHPeer is customisable and supports diverse high-level applications with a set of abstractions that are open to many possible implementations. Second, it develops an analytic hierarchy process based multi-criteria ranking approach, AHP-MCR, to rate information and help users in finding relevant items. AHP-MCR takes user context information into account. A general and extendable criteria hierarchy model is developed. The weights of the contexts criteria can be assigned by user or automatically adjusted via individual-based and/or group-based assignment. Third, it develops a Bayesian Network (BN) based user profiling method to model user’s preference. The BN model construction process is defined as being capable of handling the cold-start issue and can be applied in multiple applications. Finally, it designs and implements a Proactive Personalised News recommender, PPNews, on top of JHPeer framework. All JHPeer components are implemented in PPNews for effective news recommendation. The BN-based user profiling method estimates users’ preference including new users. The AHP-MCR approach effectively ranks news articles based on the user’s preference, past click history and news attributes. The experimental results show that PPNews can proactively recommend relevant news to mobile users.
Declaration

Whilst registered as a candidate for the above degree, I have not been registered for any other research award. The results and conclusions embodied in this dissertation are the work of the named candidate and have not been submitted for any other academic award.
Acknowledgements

I would like to extend my sincere gratitude to everyone who contributed to this research. First and foremost, I would like to express my appreciation to my supervisor, Dr. Linda Yang, for her guidance during my Master, and PhD research. She gave me the unique opportunity to work with her by allowing me to start my doctoral study in University of Portsmouth. Her invaluable directions, encouragement, advice and inspiration motivated me throughout the project. I feel extremely lucky to have been in the unique position of working under her guidance.

I would like to thank Dr. David Ndzi for reading my papers, and providing sound recommendations and consistent encouragement. I appreciate his faith in me throughout my studies and for offering me the wonderful opportunity to work as an assistant in the department’s computer programming surgery. Meanwhile, I would like to thank Dr. Mo Adda and Dr. Branislav Vuksanovic for being my examiners of PhD major review, who provided many constructive advices for this research. Mrs. Rekha Hadi deserves special thanks for proof reading my thesis.

I am also very grateful to Dr. Misha Filip, the head of School of Engineering for providing research resources, and funding conference trips. Furthermore, Linda Janes, the research administrator, has always been supportive and encouraged me during my doctoral study. A special appreciation goes to ORSAS community for offering me a much desired scholarship, which assisted me to complete PhD degree from a financial aspect. My gratitude also goes to all my colleagues in the office, Said Azzam, Somboon Toautachone, Jarupa Vipoopinyo, Taiwo Ayodele, Nurul Bostanudin. I am glad to have them as personal friends. They have been the source of some beautiful memories from the life I lived over the past four years.

Additionally, I would like to thank my friends in Hong Kong and China, especially, Ho Yin Wong, for sparking my interest in the field of computing and his friendship through the years. I want to express my gratefulness for his support on the research paper resources in the earlier stage of my study. Thank also goes to my friend Ming
Yip Lam, who helped me to proof read some of my conference papers, and Kan Xu, who brought a lot of joy to my study life aboard.

My deepest appreciation would go to my family, especially my wife, for their endless love and unconditional support. Over the years, they did more for me than I can ever pay back. Without their care and encouragement, this thesis would not have been completed.

Finally, thanks to all those who provided immense support, valuable advice, and inspired me through the journey of my PhD.
Publication

# Table of Content

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>ABSTRACT</td>
<td>I</td>
</tr>
<tr>
<td>DECLARATION</td>
<td>II</td>
</tr>
<tr>
<td>ACKNOWLEDGEMENTS</td>
<td>III</td>
</tr>
<tr>
<td>PUBLICATION</td>
<td>V</td>
</tr>
<tr>
<td>TABLE OF CONTENT</td>
<td>VI</td>
</tr>
<tr>
<td>GLOSSARY</td>
<td>X</td>
</tr>
<tr>
<td>LIST OF FIGURES</td>
<td>XI</td>
</tr>
<tr>
<td>LIST OF TABLES</td>
<td>XIII</td>
</tr>
<tr>
<td><strong>CHAPTER 1</strong> <strong>INTRODUCTION</strong></td>
<td>1</td>
</tr>
<tr>
<td>1.1 Motivation</td>
<td>1</td>
</tr>
<tr>
<td>1.2 Research Goals</td>
<td>3</td>
</tr>
<tr>
<td>1.3 Major Contributions</td>
<td>3</td>
</tr>
<tr>
<td>1.4 Thesis Organisation</td>
<td>5</td>
</tr>
<tr>
<td><strong>CHAPTER 2</strong> <strong>BACKGROUND AND RELATED WORK</strong></td>
<td>6</td>
</tr>
<tr>
<td>2.1 Mobile Information Retrieval</td>
<td>6</td>
</tr>
<tr>
<td>2.1.2 Applications of Mobile Information Retrieval</td>
<td>8</td>
</tr>
<tr>
<td>2.2 Context-Aware Computing</td>
<td>10</td>
</tr>
<tr>
<td>2.2.1 What is Context?</td>
<td>10</td>
</tr>
<tr>
<td>2.2.2 Context-Aware System</td>
<td>11</td>
</tr>
<tr>
<td>2.2.3 Context-Aware Framework</td>
<td>13</td>
</tr>
<tr>
<td>2.2.4 User Profiling Techniques</td>
<td>16</td>
</tr>
<tr>
<td>2.3 Information Recommendation for Mobile Users</td>
<td>18</td>
</tr>
<tr>
<td>2.3.1 Recommendation Techniques</td>
<td>18</td>
</tr>
<tr>
<td>2.3.2 Multi-Criteria Ratings</td>
<td>21</td>
</tr>
<tr>
<td>2.3.3 Cold-Start Problem</td>
<td>25</td>
</tr>
<tr>
<td>2.3.4 Mobile Recommendation Systems</td>
<td>26</td>
</tr>
<tr>
<td>2.4 Summary</td>
<td>28</td>
</tr>
</tbody>
</table>
CHAPTER 3  JHPEER FRAMEWORK .................................................................29

3.1  JHPEER FRAMEWORK ........................................................................30
  3.1.1 Mobile Peer .....................................................................................31
  3.1.2 Super Peer .......................................................................................32

3.2  NETWORK LAYER ..................................................................................32
  3.2.1 Network Topology ...........................................................................33

3.3  JHPEER ENGINE ....................................................................................34
  3.3.1 Context Service ................................................................................34
  3.3.2 Caching/Indexing Components ..........................................................45
  3.3.3 Query Service ..................................................................................46
  3.3.4 Peer Monitor ...................................................................................47
  3.3.5 3rd Party Services ............................................................................48

3.4  APPLICATION LAYER ............................................................................48
  3.4.1 Application Scenarios ......................................................................49

3.5  USER PROFILING ................................................................................50
  3.5.1 Bayesian Network in User Profiling .................................................51
  3.5.2 Bayesian Network Advertisement .....................................................54

3.6  SUMMARY ............................................................................................56

CHAPTER 4  ANALYTIC HIERARCHY PROCESS BASED MULTI-CRITERIA
  RANKING APPROACH .............................................................................58

4.1  MOTIVATION ........................................................................................58

4.2  AHP-MCR APPROACH .........................................................................58
  4.2.1 Construction of Criteria Hierarchy and the Alternatives .................60
  4.2.2 Construction of Criteria Comparison Pairwise Matrix .....................62
  4.2.3 Derivation of Weight of Criteria from Comparison Pairwise Matrix .....63
  4.2.4 Computation of Priority for Each Alternative .................................64

4.3  APPLICATIONS OF AHP-MCR APPROACH ....................................64
  4.3.1 Movie Domain ................................................................................65
  4.3.2 Tourism Domain ............................................................................66

4.4  SUMMARY ...........................................................................................68

CHAPTER 5  DESIGN AND IMPLEMENTATION OF A PROACTIVE CONTEXT-
  AWARE NEWS RECOMMENDER ..............................................................69
<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
</tr>
</thead>
<tbody>
<tr>
<td>5.1</td>
<td>INTRODUCTION</td>
</tr>
<tr>
<td>5.2</td>
<td>SYSTEM OVERVIEW</td>
</tr>
<tr>
<td>5.3</td>
<td>IMPLEMENTATION OF JHPEER COMPONENTS</td>
</tr>
<tr>
<td>5.3.1</td>
<td>Implementation of Context Service</td>
</tr>
<tr>
<td>5.3.2</td>
<td>Caching Service</td>
</tr>
<tr>
<td>5.3.3</td>
<td>Query Service</td>
</tr>
<tr>
<td>5.3.4</td>
<td>Peer Monitor</td>
</tr>
<tr>
<td>5.4</td>
<td>IMPLEMENTATION OF NEWS SERVICE</td>
</tr>
<tr>
<td>5.4.1</td>
<td>News Documents</td>
</tr>
<tr>
<td>5.4.2</td>
<td>News Service Protocol</td>
</tr>
<tr>
<td>5.4.3</td>
<td>RSS Aggregator</td>
</tr>
<tr>
<td>5.5</td>
<td>NEWS RANKING</td>
</tr>
<tr>
<td>5.5.1</td>
<td>Construction of Criteria Hierarchy, and the Alternative</td>
</tr>
<tr>
<td>5.5.2</td>
<td>Construction of Criteria Comparison Pairwise Matrix</td>
</tr>
<tr>
<td>5.5.3</td>
<td>Derivation of Weight of Criteria from Comparison Pairwise Matrix</td>
</tr>
<tr>
<td>5.5.4</td>
<td>Computation of Priority for Each Alternative</td>
</tr>
<tr>
<td>5.6</td>
<td>USER INTERFACE</td>
</tr>
<tr>
<td>5.7</td>
<td>SUMMARY</td>
</tr>
<tr>
<td>6.1</td>
<td>EXPERIMENTAL SETUP</td>
</tr>
<tr>
<td>6.1.1</td>
<td>Experiment Result</td>
</tr>
<tr>
<td>6.1.2</td>
<td>Conclusion/Discussion</td>
</tr>
<tr>
<td>6.2</td>
<td>EVALUATION OF AHP-MCR APPROACH</td>
</tr>
<tr>
<td>6.2.1</td>
<td>Evaluation of Ranking Performance</td>
</tr>
<tr>
<td>6.3</td>
<td>USER EXPERIENCE</td>
</tr>
<tr>
<td>6.4</td>
<td>DISCUSSION</td>
</tr>
<tr>
<td>6.5</td>
<td>SUMMARY</td>
</tr>
<tr>
<td>7.1</td>
<td>CONCLUSIONS</td>
</tr>
<tr>
<td>7.2</td>
<td>FUTURE WORK</td>
</tr>
<tr>
<td>REFERENCES</td>
<td></td>
</tr>
</tbody>
</table>
APPENDIX A  LIST OF JHPEER CONTEXT PROVIDER ........................................ 126
APPENDIX B  JHPEER COMPONENT CLASS DIAGRAMS .................................... 128
APPENDIX C  JHPEER XML SCHEME ............................................................... 131
APPENDIX D  QUESTIONNAIRE ...................................................................... 137
# Glossary

The following abbreviations are used throughout this thesis:

<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AHP</td>
<td>Analytic Hierarchy Process</td>
</tr>
<tr>
<td>AHP-MCR</td>
<td>Analytic Hierarchy Process based Multi-Criteria Ranking</td>
</tr>
<tr>
<td>API</td>
<td>Application Programming Interface</td>
</tr>
<tr>
<td>BN</td>
<td>Bayesian Network</td>
</tr>
<tr>
<td>CARS</td>
<td>Context-Aware Recommendation System</td>
</tr>
<tr>
<td>CBF</td>
<td>Content-based Filtering</td>
</tr>
<tr>
<td>CF</td>
<td>Collective Filtering</td>
</tr>
<tr>
<td>CPT</td>
<td>Conditional Probability Table</td>
</tr>
<tr>
<td>GPS</td>
<td>Global Positioning System</td>
</tr>
<tr>
<td>IDF</td>
<td>Inverse Document Frequency</td>
</tr>
<tr>
<td>IR</td>
<td>Information Retrieval</td>
</tr>
<tr>
<td>JHPeer</td>
<td>JXTA-based Hybrid Peer-to-peer</td>
</tr>
<tr>
<td>MAE</td>
<td>Mean Absolute Error</td>
</tr>
<tr>
<td>MCDM</td>
<td>Multi-Criteria Decision Making</td>
</tr>
<tr>
<td>P2P</td>
<td>Peer-to-Peer</td>
</tr>
<tr>
<td>PPNews</td>
<td>Proactive Personalised News</td>
</tr>
<tr>
<td>RSS</td>
<td>Really Simple Syndication</td>
</tr>
<tr>
<td>TF</td>
<td>Term Frequency</td>
</tr>
<tr>
<td>XML</td>
<td>eXtensible Markup Language</td>
</tr>
</tbody>
</table>
List of Figures

Figure 3.1 JHPeer Framework ........................................................................................................... 30
Figure 3.2 JHPeer Network Topology ............................................................................................... 33
Figure 3.3 Relationship between Context Service and Other Components ..................................... 35
Figure 3.4 Context Object Diagram .................................................................................................. 39
Figure 3.5 Ontology Model using JHPeer Context Representation .................................................. 43
Figure 3.6 Context Service Sequence Diagram .................................................................................. 45
Figure 3.7 Multiple Applications accessing JHPeer Engine in Modern Mobile Platform .................. 49
Figure 3.8 XML Schema of Bayesian Network Advertisement .......................................................... 55
Figure 4.1 General Recommendation Criteria Hierarchy .................................................................... 60
Figure 5.1 Information Flow between Mobile Peer and Super Peer ................................................... 70
Figure 5.2 XML Schema of NewsUserSetting .................................................................................... 73
Figure 5.3 XML Schema of NewsUserProfile ................................................................................... 74
Figure 5.4 XML Schema of FeedChannel .......................................................................................... 74
Figure 5.5 XML Schema of FeedItem ................................................................................................ 75
Figure 5.6 XML Schema of FeedScore .............................................................................................. 75
Figure 5.7 News Context Model ......................................................................................................... 76
Figure 5.8 XML Schema of FeedOperationMessage ........................................................................ 78
Figure 5.9 PPNews AHP Hierarchy ..................................................................................................... 79
Figure 5.10 Prior Probability Distributions of News BN ..................................................................... 81
Figure 5.11 Posterior Probability Distributions .................................................................................. 82
Figure 5.12 Posterior Probability Distributions with Soft Evidence .................................................. 83
Figure 5.13 Time Recency .................................................................................................................. 86
Figure 5.14 PPNews Screenshots on BlackBerry, Android and Standard Java Emulators ................. 91
Figure 5.15 Main UI on Android Phone ............................................................................................. 92
Figure 5.16 News Wizard UIs ............................................................................................................. 93
Figure 5.17 News Category (left), List of News (middle), and News Display (right) ......................... 93
Figure 6.1 PPNews Memory Usage ................................................................................................... 96
Figure 6.2 Number of Received News Articles vs Battery (HTC Desire) ..................98
Figure 6.3 Number of Received News Articles vs Battery (Google Galaxy Nexus) ....98
Figure 6.4 Received KB in Each 10 minutes (HTC Desire) ........................................100
Figure 6.5 Received KB in Each 10 minutes (Google Galaxy Nexus) .......................100
Figure 6.6 Effectiveness and Accuracy .......................................................................107
Figure 6.7 Efficiency .....................................................................................................107
Figure 6.8 Usability and Layout ..................................................................................108
Figure 6.9 Connectivity, Stability and Reliability .......................................................109
List of Tables

Table 2.1 Summary of Criteria Factors used in Existing Recommendation Systems. 24
Table 4.1 Example of Movie Domain Criteria Hierarchy .............................................. 66
Table 4.2 Example of Tourism Domain Criteria Hierarchy ........................................... 67
Table 5.1 Sample Dataset for CPM Computation ............................................................ 87
Table 5.2 Computation of Numerical Judgement of Criteria ......................................... 88
Table 5.3 Construction of CPM ........................................................................................ 89
Table 5.4 Computation of Local Score of News Articles ................................................. 90
Table 5.5 Computation of Final Score ............................................................................. 90
Table 6.1 Total Data Transmitted during Battery Lifespan ............................................. 100
Table 6.2 Average Time Used in System Operation (HTC Desire) ................................. 101
Table 6.3 Average Time Used in System Operation (Google Galaxy Nexus) ............... 101
Table 6.4 Experimental Results of MAE ........................................................................ 104
Table 6.5 Experimental Results of Precision@Top-N .................................................... 105
Chapter 1

Introduction

Context-awareness has become an essential part in various personalised applications such as mobile recommender systems and mobile information retrieval systems. Much progress has been made in context-aware recommender systems and information retrieval. However, there is lack of a general framework to support the rapid development of context-aware applications and to enable the sharing and dissemination of context information across different applications. This dissertation investigates a hybrid approach for supporting personalised services in context-aware mobile environment.

1.1 Motivation

The development of Internet, mobile and wireless technologies has resulted in an exponential growth of information and mobile users. The huge volume of information can be overwhelming to the users, exceeding the human’s perception ability to distinguish relevant information from irrelevant. New information which is of interest to a user is published and made available in our surroundings all the time. Besides, centralised computational services can be overloaded by the increasing number of mobile users.

Mobile users are engaged in different activities than those of desktop users. They have different information needs that require different approaches to address these needs. The physical and social context of mobile users is continually changing. As their context changes, their information needs will often be changing too. With the advance of semi-conductor techniques and continuous development, the more recent mobile devices are embedded with more and more sensors such as accelerometer, proximity, ambient light, digital compass, temperature, etc. Context information provides an important basis for understanding mobile users’ information need. The notion of context awareness has received a great deal of
attention recently (J.-y. Hong, Suh, & Kim, 2009; Said, Berkovsky, & Luca, 2010). Automatic user profiling and information recommendation using these context information have the potential to deliver more relevant information to mobile users in real time without them having to construct a query directly (J. Hong, Suh, Kim, & Kim, 2009). Proactive context-aware recommender (N. Chen & Chen, 2010) is a potential solution to overcome information overload and common limitations of mobile devices such as inconvenience of data input and Internet browsing. A key challenge is to make information relevant within mobile user’s dynamic environment, and automatically push interesting information to mobile users based on users’ information needs.

Many mobile recommender systems have been developed in different domains. The typical examples include tourism (Burigat, Chittaro, & De Marco, 2005; Cena et al., 2006; van Setten, Pokraev, & Koolwaaij, 2004), restaurants (Hung-Wen & Von-Wun, 2004), movie (Ono, Kurokawa, Motomura, & Asoh, 2009), news (Abbar, Bouzeghoub, & Lopez, 2009; C.-Y. Wang, Wu, & Chou, 2010; Woerndl, Brocco, & Eigner, 2009; Yap, Tan, & Pang, 2007), etc.

However, most of these mobile recommenders share four common limitations.

- Firstly, these systems focus on providing information based on explicit user query (Cena et al., 2006; Hung-Wen & Von-Wun, 2004; H. J. Lee & Park, 2007). This propounds a lot of problems on mobile devices.
- Secondly, the system architecture is specifically designed and built for particular type of context-aware recommendation such as tourism, movie, news, etc., which limits the extendibility of the recommender in other application domains.
- Thirdly, context-aware recommendation systems incorporate similar contextual information such as time, location, user interest, etc. For example, NAMA context-aware reminder (Kwon, Choi, & Park, 2005) reminds users about important tasks based on user preference, time and location contexts. Zheng (2010) designed an activity recommendation system using users’ history with GPS logs. These context-aware systems use
the common context information such as location, operation history, preferences, etc. to provide personalised information. The context information can be shared between services. In addition, context-aware systems use similar components to enable context-aware computing such as context acquisition, context retrieval, etc. However, previous studies have organised the collected context data in various structures which are difficult to share between context-aware services.

- Fourthly, most of these context-aware recommenders are considered as single-criteria recommendation which utilise user profiles/preferences, or compute score of items by simple weighted sum. Even though some studies (Adomavicius, Sankaranarayanan, Sen, & Tuzhilin, 2005; Manouselis & Costopoulou, 2007a; Yin Zhang, Zhuang, Wu, & Zhang, 2009) apply multi-criteria recommendation, they fail to change and adopt the criteria structure for different situations.

This dissertation tries to make good progress in addressing the above challenges.

1.2 Research Goals

The goal of this dissertation is to

- develop a generic framework for supporting context distribution and easing the development of context-aware applications;
- support proactive recommendation using multi-criteria ranking; and
- demonstrate the feasibility of the proposed approaches in real applications.

1.3 Major Contributions

This dissertation makes the following primary contributions:

- A context-aware framework. It systematically develops a JXTA-based Hybrid Peer-to-peer framework (JHPeer) for supporting context distributions and personalised services in context-aware mobile environment. JHPeer supports the acquisition, representation, storing,
indexing, sharing and delivery of context information and provide modular components that are common across applications. Third party components can be added to extend the functionality. This framework includes techniques, tools and application programming interfaces (APIs) to facilitate the development of context-aware applications. JHPeer consists of two types of peers: Mobile Peer and Super Peer. Both the Mobile Peer and Super Peer are embedded with the fundamental components to support context-aware applications such as Context Service, Caching/Indexing Components, Query Service, etc.

- **A multi-criteria ranking approach.** An Analytic Hierarchy Process based Multi-Criteria Ranking approach (AHP-MCR) is developed to rate information and help users finding items they prefer. AHP-MCR takes account of user context information. A general and extendable criteria hierarchy model is developed. The weights of the contexts (criteria) can be assigned by user or automatically adjusted via individual-based and/or group-based assignment.

- **A user profiling method.** JHPeer provides a Bayesian Network (BN) based user profiling method to model user’s preference through users’ historical activities and contexts. The BN model construction process is defined being capable of handling the cold-start problem and can be applied in multiple applications.

- **Design and implementation of a mobile recommender.** It develops a Proactive Personalised News recommender (PPNews) on top of JHPeer framework. PPNews is a concrete case study of JHPeer framework, where all JHPeer components are implemented and coordinated for news recommendation. The BN-based user profiling method is applied to estimate users’ preference. The AHP-MCR approach effectively ranks news articles based on a user’s preference, past click history and news attributes. PPNews is tested and evaluated in real conditions. The experimental results show that PPNews can proactively recommend relevant news to mobile users.
1.4 Thesis Organisation

The remainder of the thesis is organised as follows. Chapter 2 introduces the research background and surveys related work. Chapter 3 presents the JXTA-based Hybrid Peer–to-peer framework, details its mechanism and implementation layers and describes the design of each component. The Bayesian Network based user profiling method is discussed. Chapter 4 introduces the analytic hierarchy process based multi-criteria ranking approach. Chapter 5 presents the design and implementation of PPNews. Chapter 6 presents the results from experiments. This thesis concludes in Chapter 7 by summarizing the thesis and pointing the future research directions.
Chapter 2

Background and Related Work

This chapter presents the background and related work on mobile information retrieval and context-aware information recommender systems. It discusses how this research relates to and differs from previous research works by others. Section 2.1 provides an overview of the background on mobile IR. Section 2.2 presents existing context-aware techniques. Section 2.3 reviews existing recommendation techniques and discusses the limitations of current recommendation methods.

2.1 Mobile Information Retrieval

Mobile Information Retrieval is a subset of traditional information retrieval that aims to locate relevant information for the dynamic mobile environment and adopts the information content to fit on mobile devices (Flora, 2010). Traditional information retrieval techniques were developed to achieve efficient document retrieval over a large collection of documents. These later on, started to focus towards more hybrid and distributed information systems such as the World Wide Web (WWW). Contemporary web search engines and web directories target desktop users by returning them documents based on user queries. Similar access to existing WWW contents is not enough for a mobile user.

With the realisation of pervasive computing technologies (Romero et al., 2010; Tan, Sheng, Wang, & Li, 2010), user surrounding is becoming information rich and is playing a more significant role. In such an environment, a typical mobile user will come across different information sources while being on the move. The nomadic nature of user puts a lot of constraints and dynamics to user’s context. Mobile users’ information needs change as they change their location as well as other contextual attributes such as task, emotional or environmental context. A typical interaction scenario may include user interacting with remote information resources as well as other distributed information sources or vice-versa. Such an interaction demands
unified access to dynamic and distributed information resources. Computational services need to become as mobile as their users.

With the outbreak of ad-hoc information-sharing networks (Bisignano, Di Modica, & Tomarchio, 2005b; Schifanella, Panisson, Gena, & Ruffo, 2008; Woerndl et al., 2009) in our surrounding and with the inclusion of existing information that floods the WWW, it would be impossible to index and pre-process all of these information resources to a centralised location. Hybrid solutions would have to be considered in order to tap these information sources. Mobile information retrieval is becoming an exciting domain that challenges information scientists (Flora, 2010).

2.1.1 Technological drivers

Mobile devices were mainly used to communicate with others by voice or messaging. Today, with the constant development of mobile devices, they possess as much on board connectivity equipment as most Computers do, faster CPUs, increased memory size, and in some cases, such as GPS (Global Positioning System), the mobile device exceeds the computer. Modern mobile device can report the location of a mobile user using GPS, which can be used to retrieve more relevant information based on user’s location and situation. Mobile devices, including mobile phones, PDAs and tablet PCs are capable of transferring, storing and processing a wider variety of information than just voice data. Mobile devices have intruded into our lives as multi-functional multimedia equipment that is not only for communication but also for advanced services such as mobile searching, Internet browsing, mobile dating (Ballagas et al., 2007; Kwon et al., 2005; Ruta, Di Noia, Di Sciascio, & Scioscia, 2008). According to a recent study (Anderson & Rainie, 2008), mobile devices will surpass computers as the primary tool for Internet by 2020. Mobile network operators have also acknowledged this move towards mobile services by providing unlimited Internet access for mobile devices (Richardson, 2006). Mobile devices have changed the way people communicate, interact in the physical world, and coordinate their social activities (Rheingold, 2003).
A wide variety of mobile applications have been developed to provide services on mobile devices. Geographic information retrieval system (Mountain & MacFarlane, 2007) was implemented in client-server architecture, which retrieves Point of Interest (POI) based on user query. The number of users is limited by the capacity of server and therefore lack scalability. UbiqMuseum (Cano, Manzoni, & Toh, 2006), a mobile tour guide system that provides information about current places or exhibited objects, uses Bluetooth-based technologies to locate and deliver information to mobile users. Multiple Bluetooth enabled accessing servers were deployed in the environment to communicate with users within 10 meters distance. This system architecture is flexible enough to handle a large number of users. However, the Bluetooth network provides limited coverage, which limits the mobility of mobile users and extensibility of services. Another drawback is Bluetooth’s power consumption for device discovery and the time needed for locating and establishing a connection with encountering devices. VANETs motor information service (Woerndl et al., 2009) attempts to recommend nearest fuel station when fuel becomes low. In this system, information is stored locally in each device and updated information is exchanged between devices or hotspots via ad-hoc connections. The level of data consistency is not guaranteed since the updating mechanism is based on items’ timestamp. It may return out-dated information to users in runtime. Mobile information search (Church, Neumann, Cherubini, & Oliver, 2010; Lane, Lymberopoulos, Zhao, & Campbell, 2010; Mario & José, 2009) provides personalised search and tries to overcome common issues of using small mobile devices on traditional web searching such as limited input and screen size. These mobile services mainly utilise user location to provide relevant information to mobile users. The above related works are mainly based on centralised approaches that suffered from single point of failure and lacked scalability. Besides, these systems are designed for specific application and the common context information is modelled in different formats which cannot be shared directly amongst applications. This work differs from all of the above works. The main difference is that it developed a hybrid architecture and a novel context-driven approach for
mobile information retrieval and sharing. JHPeer incorporates context-awareness information retrieval engine in all types of peers which uses both user and device context for effective IR in mobile environments.

In contrast to centralised applications, some mobile applications are developed in decentralised network. Gasoline Price Comparison System (Volovikov et al., 2008) collects gasoline price information when users come across gas stations and delivers gasoline prices to mobile users using mobile encounter information diffusion. As the information is accumulated, the application can suggest the cheapest gas station. Similarly, MOPNET (Wolfson, Bo, Huabei, & Hu, 2006) also delivers information such as free parking slots, traffic data, taxicabs to mobile users via wireless ad-hoc network. LightPeers (Christensen, 2007) platform is a mobile P2P application platform coded using Microsoft .Net framework. It is designed for building small sociality systems with short range group connections. Proem (Kortuem, 2002), Peer2Me (A. I. Wang, Bjornsgard, & Saxlund, 2007) and JMobiPeer (Bisignano, Di Modica, & Tomarchio, 2005a) are designed for Mobile Ad-hoc Network (MANET) environment which is not suitable for large-scale networks, and can only support Java programming language. MOBY (Horozov, Grama, Vasudevan, & Landis, 2002) is built for mobile P2P networks based on Jini Technology. It uses remote procedure call (RPC) that relies on Server Computation, which is not suitable for dynamic mobile environment.

Some P2P systems have been developed based on JXTA (Juxtapose) framework (JXTA, 2007), which supports the communication between peers and provides common P2P discovery mechanisms. JXTA is a set of protocol specifications for implementing peer-to-peer applications on a global scale with the aims of Interoperability, Platform independence, and Ubiquity (JXTA, 2007). JXTA protocols are designed to be independent of transport protocols (i.e. TCP, UDP, multicast) and establish a virtual network on top of existing physical network infrastructure (Interoperability). Besides, P2P communication can be established without referring the actual software implementations such as programming languages, device’s platforms and Platform independence, which can be connected behind firewall and/or Network Address Translation (NAT). JXTA technology is not only designed
for PCs, but can be accessible by any devices (Ubiquity). Expeerience (Bisignano, Calvagna, Modica, & Tomarchio, 2003) and JMobiPeer (Bisignano et al., 2005a) used JXTA for implementing services in MANET (Mobile Ad-Hoc network). Jadabs (Frei, 2005) designed a OSGi-based middleware platform on top of JXTA communication framework. MoSoSo (Tsai, Han, Xu, & Chua, 2009) is a P2P mobile social software that enables mobile users to share, discover, communicate resources with each other.

All the above systems are similar to the peer communication in JHPeer in that all of them are built on top of JXTA framework. Unlike JHPeer, they do not provide context management for development of shared context-awareness applications.

2.2 Context-Aware Computing

Context-aware computing evolved from the idea of ubiquitous computing which was introduced by Mark Weiser (Weiser, 1999) in the 1990s. The idea refers to seamless integration of devices into a user’s everyday life and offers anytime, anywhere, anyone computing by decoupling users from devices (Bardram, Hansen, Mogensen, & Soegaard, 2006; Dey, 2001; Hill et al., 2004; J.-y. Hong et al., 2009; Schilit & Theimer, 1994). Salber, Dey, and Abowd (1999) indicated that a system is context-aware if it uses context to provide relevant information and services to the user. The notion of context awareness in information retrieval has received a great deal of attention recently (Bahrami, Yuan, Smart, & Shadbolt, 2007; Baldauf, Dustdar, & Rosenberg, 2007; J.-y. Hong et al., 2009). By exploiting contextual information of the system, and of users in particular, information retrieval experience of a user can be enhanced and the overall performance of information retrieval process can be significantly improved.

2.2.1 What is Context?

Numerous definitions of context exist. Lexical meaning of the term “context” defines it as the set of facts or circumstances that surround a situation or event. The widely accepted definition of context in context-aware computing (Baldauf et
al., 2007; Gu, Pung, & Zhang, 2005; Mostéfaoui, 2004) is defined by Dey and Abowd (1999) as follows:

"Context is any information that can be used to characterise the situation of an entity. An entity is a person, place, or object that is considered relevant to the interaction between a user and an application, including the user, and applications themselves."

Typically, Context in context-aware system can be divided into two major types (Henricksen, Indulska, & Rakotonirainy, 2002): Static Context, and Dynamic Context. Static Context is seldom changing during system operation such as personal calendar/schedule, address book, contact list, to-do list, user profile/preferences and hardware profile, etc. Dynamic Context is highly variable information such as user location, weather, temperature, time, speed, system status, nearby resources/friends, short term user interests and user emotion, etc.

2.2.2 Context-Aware System

Context aware systems sense and adapt their behaviour based on the changing contexts (Bolchini, Schreiber, & Tanca, 2007; Dey & Abowd, 1999; J. Lee & Lee, 2007), which generally consists of four basic components (Baldauf et al., 2007): Context Acquisition, Context Discovery, Context Model, and Context Processing.

2.2.2.1 Context Acquisition

Context Acquisition refers to process of obtaining user context information. H. L. Chen (2004) states that context information can be obtained by various methods such as directly accessing hardware sensors, facilitating a middleware infrastructure and acquiring context from a context server. Direct access to hardware sensor means the high-level application can directly obtain data from low-level device sensors such as GPS sensor, temperature sensor, battery sensor, etc. Being facilitated by a middleware infrastructure refers to implementing a service layer that acts as a bridge between low-level hardware interface and high-level application. It provides scalable and reusable context acquisition modules. It hides
the complexity of low-level programming and simplifies the implementation of different hardware sensors for application developers. Acquiring context from a context server means context information is provided by server-based computation rather than acquiring contexts from device itself. The idea is to distribute and reuse available context information from context server. This approach permits low-end mobile devices to acquire context information even when it is not embedded with hardware sensors.

Much research effort has been made on capturing context information by software systems and user input (H. L. Chen, 2004; Gu et al., 2005; J. Hong et al., 2009; Indulska & Sutton, 2003; Padovitz, Loke, & Zaslavsky, 2008). These systems can capture context directly from local devices, sensors or digital tags (i.e. RFID, NFC tags). JHPeer achieves the same goal but also improves the process through searching and sharing context information over peer networks.

2.2.2.2 Context Discovery

Context Discovery, or so called Resource Discovery is the mechanism to locate and find available sensors in distributed networks. Context Toolkit (Dey, Abowd, & Salber, 2001) implemented a context discoverer which contains registry components to keep a list of current sensors. It also contains a “ping” mechanism to check the availability of sensors and remove any unavailable sensors from the registry. Similar registry and discovery approaches have been used in Gaia (Manuel, 2002), SOCAM (Gu et al., 2005). JHPeer reuses the discovery mechanism in JXTA.

2.2.2.3 Context Modelling

Context Model has to be defined in order to store, share, transmit and manipulate context in the system. There are several approaches for context modelling (Bettini et al., 2010; Strang & Linnhoff-Popien, 2004) such as Key-Value models, Markup (XML) schema models, Graphical (UML) models, Object-oriented models, Logic-based models, and Ontology-based (RDF/OWL) models. However, these context models are inflexible and not exchangeable. Researchers have noticed the problem
and moved on to hybrid models enabling context model convert (Achilleos, Yang, & Georgalas, 2010; Bettini et al., 2010).

JHPeer uses an abstract context representation that allows encapsulation of different types of textual model that enable context distribution.

### 2.2.2.4 Context Processing

Context Processing includes: pre-processing, post-processing and reasoning. The pre-processing and post-processing are optional tasks that are used to filter noise contexts, notify components when context changes or handle missing context attributes from sensors. Context Reasoning is the main task that deduces new and relevant information to the use of applications from the various sources of context-data. The common approaches to carry out context reasoning includes: Bayesian Network (Anand, 2004; Hwang & Cho, 2009; Marco, 2007), case-based (Kofod-Petersen & Mikalsen, 2005), logic-based (Ranganathan & Campbell, 2003), ontology-based (Joung & Chuang, 2009; Shen & Yang, 2004; X. H. Wang, Zhang, Gu, & Pung, 2004) and rule-based (Bikakis, Patkos, Antoniou, & Plexousakis, 2008; Y. Yu, Kim, Shin, & Jo, 2009). The implementation of Context Reasoning approach is related to the application domains.

### 2.2.3 Context-Aware Framework

The context-aware framework is usually used as the system infrastructure for developing context-aware applications, which combines context components, data storage, network communication modules and any other system modules based on design criteria. There are three design approaches for context-aware framework: Server-based, Distributed and Hybrid approaches.

Server-based context-aware framework uses client-server architecture. The applications are running on a centralised server, where the client is usually a thin client such as a mobile device. Most of the computation will be carried out on the server. Several frameworks like Context Toolkit (Salber et al., 1999) and CoBrA (H. L. Chen, 2004) have been proposed to implement server-based context-aware
applications. SOCAM (Gu et al., 2005) is ontology-based context-aware service focused on a smart space domain. It uses OWL Web Ontology Language to resolve the issues of context modelling and context reasoning. However, it is implemented in Java RMI technology and does not provide facility for ranking/making recommendations. CASS (Fahy & Clarke, 2004) is a server-based middleware that supports context-aware application on mobile devices. The middleware supports the use of large numbers of context inputs. JHPeer is based on hybrid architecture that supports more services in shared platforms and accommodates heterogeneous context sources. Distributed context-aware frameworks use peer-to-peer architecture and allow the management of context information in a distributed environment. Several groups have tried to implement distributed context-aware applications. CMF (van Kranenburg, Bargh, Iacob, & Peddemors, 2006) is designed for context management to process and exchange context information in mobile environment. It aims to operate in pure peer-to-peer architecture. EDUTELLA (Nejdl et al., 2002) introduced a P2P sharing framework which can distribute and discover context information (in RDF) in desktop environments. CARS (Abbar et al., 2009) implemented a service-oriented P2P environment that made use of discovery function and nearest-neighbours algorithm. These approaches are not suitable for mobile devices since it requires mobile peers to handle numerous queries from other peers and may drain the battery.

Hybrid approach takes the advantage of both Server-based and pure P2P frameworks and is more flexible and scalable. DYNAMOS (Riva & Toivonen, 2007) proposes a hybrid model of context-aware service provisioning. It supports client-to-client communication in addition to client-to-server communication for information sharing. Mobile users can subscribe for weather services to receive real-time weather information or subscribe for nearby user notes to receive user comments. However, it is designed for specific mobile platform (i.e. Symbian OS) and does not provide a context model. ECORA hybrid framework (Padovitz et al., 2008) focuses on context reasoning. Context information is modelled based on ontology. Similar to JHPeer, Hu et al. (2010) proposes a hybrid P2P context distribution system based on JXTA. Three types of Peers are defined: Sensor peers
provide context information; Disseminator peers process and store context information; Consumer peers consume context information from sensor peers and disseminator peers. However, this work is just at the beginning. Context management is not provided and implementation of proposed approach is still under development.

The above related works manage contextual information in centralised services, pure p2p fashion, or hybrid approaches. The centralised approach suffers from a single point of failure and has limited scalability while the distributed approach may overload the mobile devices and is less efficient due to the rapid growing number of mobile users seeking dynamic information. The hybrid approach improves the flexibility and scalability of context-aware framework. However, majority of the existing hybrid frameworks have not been updated (i.e. bug fix, adopt modern smartphone platform). Modern mobile devices are capable of storing huge amounts of information such as context Information and user documents. It could become a significant information source. JHPeer framework differs from the above works for four reasons. First, it includes optional indexing and query components in addition to simple data repository on mobile devices. Second, information can be shared through distributed data repository or stored in local data repository, which avoid potential bottlenecks. Mobile Peers can communicate directly with each other. Third, JHPeer framework provides abstraction APIs for development of context-aware systems that allow developers to implement the specific functions or algorithms for particular applications. For rapid development, JHPeer also provides basic implementation of defined components. Fourth, the components are reusable throughout JHPeer framework. Third party components can be integrated in JHPeer framework in order to extend the functionality. Each JHPeer component is discussed in details in section 3.3.

2.2.3.1 Context caching and indexing

Context Caching refers to storing the collected contextual information in data storage or database. CMF (van Kranenburg et al., 2006) and SOCAM (Gu et al., 2005) used relational databases to store RDF-based contexts and used SQL to retrieve it.
CASS (Fahy & Clarke, 2004) CoBrA (H. L. Chen, 2004), Hydrogen (Thomas, 2003) and Hong (J. Hong et al., 2009) stored contextual information in server side only. The majority of the context-aware frameworks only provide caching and/or indexing facility for specific type of context model and mainly on server side. However, the modern mobile devices have relatively large storage. Information can be stored for the purpose of exchanging information between peers or handling any retrieval query. Unlike these context-aware frameworks, JHPeer provides APIs to cache and index contextual information in both Mobile Peer and Server Peer in order to retrieve the cached contextual information efficiently. In addition, different types of caching can be implemented in the same time. For instance, while a global cache is running to distribute and store contextual information in network peers, a local cache is running to hold user profile in local machine.

2.2.4 User Profiling Techniques

User Profiling refers to the techniques to generate a user profile for a particular user. User Profile is a collection of personal data about a specific user. It can include user interests, likes/dislikes, user knowledge, user backgrounds, user characteristics, user behaviour, and user context. The context information of a user can be logged and stored as user profile and used as long-term user interests (Bettini et al., 2010; J. Hong et al., 2009).

The User Profile (i.e. user preference) can be inputted by user via user interface or learns from user behaviours/contexts. User can provide clear instruction as to what the user is interested in and how the system should behave such as interval to retrieve information from server. It is also known as explicit profiling. For example, NewsAgent (Godoy, Schiaffino, & Amandi, 2004) provided an interface to enable users to report the interested content of news, select interested newspaper and/or news categories, and rate the news.

In contrast, implicit profiling learns user profile by monitoring user behaviour without interfering with users as they go about their operations. There are some common content-based user profiling techniques in information services like
information retrieval and filtering system, recommendation system. Keywords-based techniques generate user profile by harnessing user history to determine user interested keywords. Grčar, Mladenič, and Grobelnik (2005) use TF-IDF algorithms of user browsing history to construct a user profile with user short-term and long-term interests. Similarly, Lieberman, Fry, and Weitzman (2001) and NewsDude (Billsus & Pazzani, 1999) also used this technique to model user interests. Additionally, NewsDude (Billsus & Pazzani, 1999) used a Bayesian Classifier to analyse whole article to determine user long-term interests.

Bayesian Network (BN) is inference and representation method for modelling user needs, goals, and preferences (Horvitz, Breese, Heckerman, Hovel, & Rommelse, 1998). García, Amandi, Schiaffino, and Campo (2007) presented a work using BN to model students characteristics and learning styles in a web-based education system. The BN model is constructed based on a previous learning framework in order to identify the type of learning styles. The probability table is simply obtained by analysing students’ log files. Once the BN is constructed, it is used to determine the learning style of a student in order to provide relevant trainings. Lumiere project (Horvitz et al., 1998) in Microsoft Research used BN to model users’ goals and needs. It determines the users’ needs by considering user’s background, actions and queries. Ono et al. (2009) constructed a user preference model for movie recommendation. Different to others, it constructs the BN model with additional contextual information including location, mood, and with who. The probability table is obtained from questionnaire survey.

Moreover, there are many other user profiling techniques. J. Hong et al. (2009) applied association rule to create user preference models. The methods require experts to define the rules and tend to produce huge amount of rules. J. Lee and Lee (2007) used case-based reasoning techniques to develop a music recommender. This recommender will deliver personalised music recommendation based on previous experiences (i.e. similar user profile and/or similar situation). YourNews (Ahn, Brusilovsky, Grady, He, & Syn, 2007) builds user profile using relevance feedback algorithm, known as Rocchio’s algorithm (Baeza-Yates & Ribeiro-Neto,
1999) from Information Retrieval. Users are allowed to rate on the news to indicate likes or dislikes.

The above works focuses on generating a single user profile. Unlike others, the proposed approach divides the user profile into different sub-modules. Each sub-module can be built by different methods such as Bayesian Network, TF-IDF. In addition, most of the methods require large amounts of training data, which is not available in many cases. This thesis proposes a strategy using Bayesian Network to resolve the cold-start problem.

2.3 Information Recommendation for Mobile Users

Recommender system is an information filtering system that attempts to recommend information items or services that are likely to be of interest to the user or be relevant to user’s needs (Resnick & Varian, 1997). Typically, a recommender system predicts the rating of an information item based on a user profile. Once the rating of items (which have/have not been seen by the user) is estimated, it will recommend the top-k item(s) with highest ratings to the user, known as Recommendation. Formally, the recommendation process is defined as:

$$\forall c \in C, k_c^* = \arg \max_{k_c \in K} f^c(C_{c \in C}, k_c)$$

where $C$ is the set of all users or the subset of all users, $c$ is any user in the user space, $K$ is the set of all items and $k_c$ is the set of all possible items that can be recommended to user $c$. Then a utility function $f$ is applied to measure the usefulness of item $k_c$ to user $c$. Finally, item with highest rating will be selected and recommended to user $c$.

2.3.1 Recommendation Techniques

The recommendation (prediction) techniques are generally categorised into three types (Adomavicius & Tuzhilin, 2005; Burke, 2002; Manouselis & Costopoulos, 2007a): Content-based techniques (Cantador, Bellogín, & Castells, 2008), Collaborative techniques (Das, Datar, Garg, & Rajaram, 2007) and Hybrid approaches (D.-R. Liu, Lai, & Lee, 2009).
2.3.1.1 **Content-based Techniques**

Content-based techniques, also known as content-based filtering (CBF) is a recommendation method that will recommend items similar to the ones that the user preferred in past. Usually, content-based recommendation systems generate a user profile through user’s historical contents. The new items are then matched against the user profile to find the most relevant items for that user. Content-based technique has its root in information retrieval researches (Adomavicius & Tuzhilin, 2005). Traditional heuristics are based on information retrieval methods using Term Frequency (TF) / Inverse Document Frequency (IDF) and Vector Space Model (Bogers & Bosch, 2007; Ferman, Errico, Beek, & Sezan, 2002; Whitman & Lawrence, 2002). Other techniques have also been used such as Bayesian Classifier, clustering, decision trees, artificial neural networks (Mooney & Roy, 2000; M. Pazzani & Billsus, 1997; Yi Zhang, Callan, & Minka, 2002), etc.

2.3.1.2 **Collaborative Techniques**

Collaborative techniques, also known as collaborative filtering (CF) is a recommendation method that generates recommendations by analysing peer users’ histories and ratings. The CF algorithms are divided into two types (Breese, Heckerman, & Kadie, 1998): Memory-based algorithms and Model-based algorithms.

Memory-based algorithms use user rating data to calculate similarity between users or items and make recommendations. One of the preferred approaches is to use k-nearest neighbour (kNN) classifier to locate similar users who have similar tastes in the past (Ricci, Rokach, Shapira, & Kantor, 2010). Pearson correlation-based (Resnick, Iacovou, Suchak, Bergstrom, & Riedl, 1994; Shardanand & Maes, 1995) and cosine-based (Breese et al., 1998; Sarwar, Karypis, Konstan, & Reidl, 2001) are two popular algorithms to compute the similarity of users.

In Contrast to Memory-based algorithms, Model-based algorithms construct a predictive model to estimate the ratings of unseen items based on training data. Typically, machine learning algorithms are used such as Bayesian Network (Breese...

2.3.1.3 Hybrid Techniques

The main limitation of both CF and CBF methods is known as cold-start problem (Adomavicius & Tuzhilin, 2005; Chu & Park, 2009; Das et al., 2007). It is difficult for a recommender to understand new user’s preference and provide relevant recommendations. To address this problem, several researches have combined CBF and CF techniques and developed hybrid recommendation systems (Chu & Park, 2009; Das et al., 2007; H. J. Lee & Park, 2007). Hybrid approach can also incorporate knowledge-based techniques such as case-based reasoning (Robin, 2000) or rule based techniques (J. Hong et al., 2009). D.-R. Liu et al. (2009) demonstrates a product recommender which use CF and association rules. News@hand (Cantador et al., 2008) adopted semantic context-aware technologies to combine content-based filtering and collaborative filtering. D.-R. Liu, Tsai, and Chiu (2011) presented a blog articles recommender for mobile users. It combines three different techniques in the recommendation process: topic clustering, CF, and attention degree (click times). Topic clustering uses TF-IDF algorithms to group articles into different clusters. The attention degree predicts the popularity of the topic clusters. Finally, CF is used to predict user ratings based on previous history. The top-k articles with highest score will then be delivered to user. Moreover, Su, Yeh, Yu, and Tseng (2010) developed a context-aware music recommendation system using both CBF and CF techniques. CBF techniques are used to analyse the content of music and model it to patterns. The pattern is used to predict user interest based on user history. CF technique is used to locate music which has been listened by other users who have similar contexts in the past (i.e. location, time, light, motion, temperature, etc.).

This thesis also uses hybrid approach but distinguish it from the above based on for two reasons: First, the previous studies organise recommendation into different stages or estimated rating through a specific hybrid ranking model. AHP-MCR
approach provides a comprehensive framework that offers both ranking process and computational model. Second, these previous studies mostly utilise user profiles in the ranking process and usually consider a single-criterion value (i.e. overall rating). However, users may depend on more than one utility aspect when making their choices. AHP-MCR approach use multi-ratings for different aspects (i.e. criteria) and/or apply multi-criteria ratings algorithms in ranking process.

2.3.2 Multi-Criteria Ratings

Most of current recommender systems are classified as single-criterion rating systems (Balabanović & Shoham, 1997; Claypool et al., 1999; Konstan et al., 1997; Sarwar et al., 2001), which use a single criterion to represent an item’s utility to a user in a Users × Items space (Adomavicius et al., 2005; Cantador et al., 2008; A. Chen, 2005; Chu & Park, 2009). According to Adomavicius, Manouselis, and Kwon (2010), traditional single-criterion rating is limited because recommending items to particular user often depend on more than one utility-related aspect. The multi-criteria ratings recommendations could help to improve the quality of recommendations by providing additional information and being able to represent more complex preferences of each user (Adomavicius & Kwon, 2007; Adomavicius et al., 2010).

Recent recommender systems incorporate the multi-criteria rating information into the recommendation process. The difference between single-rating and multi-criteria rating systems is that the latter uses more information about the users and items in the recommendation process. For instance, a multi-criteria restaurants recommender (H.-H. Lee & Teng, 2007) includes four basic criteria: food, service, cost, and decoration. User can vote on each criterion. The multi-criteria recommender can calculate the predicted ratings of each criterion via Memory-based CF. Restaurant items that have closest similarity with the user and other users amongst all criteria considered will be recommended. Besides, the cost of restaurants that are over the users’ budget will be filtered.
The similarities can be computed by weighted sum (Chein-Shung, 2010; Tang & McCalla, 2009), average similarity (Adomavicius & Kwon, 2007; Poompuang & Premchaiswadi, 2010), and worst-case similarity (Adomavicius & Kwon, 2007). Equation (2.2) shows the computation of weighted sum that C is the criteria, \( w_c \) is the weight of criteria, k is the number of criteria and \( \text{similarity}_c(u, u') \) is the similarity of a particular user \( u \) and another user \( u' \) in criteria c. The average similarities are shown in equation (2.3). The worst-case similarity finds the smallest similarity in all criteria that is shown in equation (2.4).

\[
\text{similarity}_{\text{weighted sum}}(u, u') = \sum_{c=0}^{k} w_c \text{similarity}_c(u, u')
\] (2.2)

\[
\text{similarity}_{\text{avg}}(u, u') = \frac{\sum_{c=0}^{k} \text{similarity}_c(u, u')}{k}
\] (2.3)

\[
\text{similarity}_{\text{min}}(u, u') = \min_{c=0,...,k} \text{similarity}_c(u, u')
\] (2.4)

The multi-dimensional distance metrics approaches calculate the distance (dist) between two users (\( u \) and \( u' \)) on all common rated items (I), that can be written as:

\[
dist(u, u') = \frac{\sum_{i \in I(u, u')} \text{dist}(R(u, i), R(u', i))}{|I(u, u')|}
\] (2.5)

where \( |I(u, u')| \) is the common rated items of user \( u \) and \( u' \), \( \text{dist}(R(u, i), R(u', i)) \) is the distance between users on item (i) across all criteria. The distance can be calculated by using Manhattan, Euclidean, Chebyshev distance metrics (Adomavicius & Kwon, 2007). The final similarity between two users is calculated as follows:

\[
\text{similarity}(u, u') = \frac{1}{1 + \text{dist}(u, u')}
\] (2.6)

Moreover, Manouselis and Costopoulou (2007b) also proposed various methods to compute the similarity between users such as similarity-per-priority, similarity-per-evaluation, and similarity-per-partial-utility. The similarity-per-priority computes the similarity between users by comparing the weight of criteria instead of ratings. The similarity-per-evaluation computes the similarity between users using non-
weighted ratings of individual criteria that can use different users in each criterion. The similarity-per-partial-utility is similar to similarity-per-evaluation, but it uses weighted ratings of individual criteria.

Several groups have studied model based approaches. Adomavicius and Kwon (2007) proposed an abstract aggregation function that consist of 3 steps. Firstly, it computes the rating of each criterion using any recommendation techniques. Secondly, it chooses an aggregation function using domain expertise, statistical techniques, or machine learning techniques that the function can be average, weighted sum, evaluates user histories. Thirdly, the overall rating is computed by selected function. Nachiketa, Ramayya, George, and James (2008) proposed a probabilistic model which extends the flexible mixture model (FMM) (Si & Jin, 2003). Li, Wang, and Geng (2008) utilised multi-linear singular value decomposition (Lathauwer, Moor, & Vandewalle, 2000) to predict the overall rating from multi-criteria ratings via a restaurant recommendation service with 10 criteria such as cuisine, ambience, service, and so on. Yin Zhang et al. (2009) applied probabilistic latent semantic analysis to multi-criteria rating recommender using Yahoo! Movie dataset, which showed improved performance in recommend top-k items than single criterion.

Recently researchers have attempted to apply content-based approaches in multi-criteria recommendation systems. Pasi, Bordogna, and Villa (2007) and Pereira, Dragoni, and Pasi (2009) proposed a personalised multi-criteria model for information systems such as News streams, RSS feeds, and blogs. Four criteria are designed to filter documents includes aboutness, coverage, appropriateness, and reliability. The final rating is calculated by weighted sum. Wolfe and Zhang (2009) used similar approaches but used 9 different criteria on an air ticket system. Farah and Farah and Vanderpooten (2008) extended the simple aggregation mechanism with decision rules to judge the rank of documents.

The above related work on multi-criteria recommendation is complementary to this work. For example, JHPeer could use weighted sum to calculate similarity. However, these researches mainly represent all criteria in a single level in the computation
process. The number of criteria in mobile recommendation system increases dramatically. In order to deal with this complex decision problem, this thesis employ an Analytic Hierarchy Process (AHP) (Saaty, 1980) from Multi-Criteria Decision Analysis (MCDA) to decompose the criteria into a hierarchy. It reduces complex decisions to a series of one-to-one comparisons. Flexibility exists around the weight of criteria assigned via user assignment, estimated through user histories, or group decision making techniques. The criteria in AHP hierarchy can be formed using CF and/or CBF techniques. The final score of item is aggregated from the bottom to the top of criteria hierarchy.

2.3.2.1 Criteria Attributes used in Existing Studies

The context information such as personal information, context/operation history, temporal user location, weather condition, current time, item popularity, movie original country, news time recency, etc. are used in the mobile context-aware recommenders. The Criteria Factors used in existing recommendation systems is summarized in Table 2.1. This thesis identified and classified four major criteria for multi-criteria recommendation system. The four major criteria include: Profile, Situation, Rates, and Attributes (Detailed in section 4.2). Four types of variables in Profile criterion are classified for generating user profile, which include demographic variables, situational-behaviour variables, psychographic variables, and domain specific variables (Detailed in section 3.5.1).

<table>
<thead>
<tr>
<th>Factor</th>
<th>Research</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Profile</strong></td>
<td>(C.-Y. Wang et al., 2010), (J. Lee &amp; Lee, 2007), (Y. Yu et al., 2009),</td>
</tr>
<tr>
<td></td>
<td>(Ono et al., 2009), (D.-R. Liu et al., 2009), (Huang &amp; Bian, 2009),</td>
</tr>
<tr>
<td></td>
<td>(J. Hong et al., 2009), (Adomavicius et al., 2005)</td>
</tr>
<tr>
<td></td>
<td><strong>Situational-behaviour</strong></td>
</tr>
<tr>
<td></td>
<td>(J. Lee &amp; Lee, 2007), (Y. Yu et al., 2009), (Woerndl et al., 2009),</td>
</tr>
<tr>
<td></td>
<td>(Woerndl et al., 2009), (D.-R. Liu et al., 2009), (J. Hong et al., 2009)</td>
</tr>
<tr>
<td></td>
<td>(Papadogiorgaki et al., 2009)</td>
</tr>
<tr>
<td>Variables</td>
<td>2008), (Yap et al., 2007)</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>---------------------------</td>
</tr>
<tr>
<td>Psychographic variables</td>
<td>(Ono et al., 2009), (Huang &amp; Bian, 2009), (J. Hong et al., 2009)</td>
</tr>
<tr>
<td>Domain specific variables</td>
<td>(Ghosh, Mundhe, Hernandez, &amp; Sen, 1999), (J. Lee &amp; Lee, 2007), (Yin Zhang et al., 2009), (Y. Yu et al., 2009), (Woerndl et al., 2009), (Ono et al., 2009), (D.-R. Liu et al., 2009), (Huang &amp; Bian, 2009), (J. Hong et al., 2009), (Chu &amp; Park, 2009), (Papadogiorgaki et al., 2008), (Yap et al., 2007), (H. J. Lee &amp; Park, 2007), (Das et al., 2007), (Adomavicius et al., 2005), (Cantador et al., 2008)</td>
</tr>
<tr>
<td>Situation</td>
<td>(C.-Y. Wang et al., 2010), (J. Lee &amp; Lee, 2007), (Y. Yu et al., 2009), (Woerndl et al., 2009), (Ono et al., 2009), (T. Q. Lee, Park, &amp; Park, 2009), (Huang &amp; Bian, 2009), (J. Hong et al., 2009), (Chu &amp; Park, 2009), (Yap et al., 2007), (Adomavicius et al., 2005), (A. Chen, 2005)</td>
</tr>
<tr>
<td>Rates</td>
<td>(Yin Zhang et al., 2009), (Woerndl et al., 2009), (Ono et al., 2009), (D.-R. Liu et al., 2009), (T. Q. Lee et al., 2009), (Chu &amp; Park, 2009), (Yap et al., 2007), (H. J. Lee &amp; Park, 2007), (Das et al., 2007), (Adomavicius et al., 2005), (A. Chen, 2005)</td>
</tr>
<tr>
<td>Attributes</td>
<td>(C.-Y. Wang et al., 2010), (Y. Yu et al., 2009), (Woerndl et al., 2009), (Ono et al., 2009), (D.-R. Liu et al., 2009), (Huang &amp; Bian, 2009), (J. Hong et al., 2009), (Chu &amp; Park, 2009), (H. J. Lee &amp; Park, 2007), (Das et al., 2007), (Adomavicius et al., 2005), (Cantador et al., 2008)</td>
</tr>
</tbody>
</table>

### 2.3.3 Cold-Start Problem

Cold-start recommendation is a key challenge in recommender systems. In CBF recommender, new user has to rate sufficient number of items before the
recommender can determine the user’s preferences and start providing any recommendations. CF recommender often fails for new user or new items as the system has very little knowledge about the new users/items in terms of their preferences/rating.

To solve this problem, many methods have been proposed. Some researches (Ghazanfar & Prugel-Bennett, 2010; Lam, Vu, Le, & Duong, 2008; M. J. Pazzani, 1999) extended the traditional CF techniques with users’ demographic information (i.e. age, gender, occupation) to locate similar users in hybrid recommendation approaches. New user is grouped into a particular demographic segment and assumed to have similar interests. However, this method is still facing the problem of data sparsity. For example, if the user is age over 50 and the recommender does not have enough number of users in age group over 50, it will be difficult to provide reliable recommendation because the system only has very few ratings to compute the similarity between items and users.

Instead of extending traditional CF, the proposed approach uses Bayesian Network with demographic data to predict user preference to overcome the cold-start problem. Since Bayesian Network was originally designed for reasoning uncertainty, Bayesian Network is also used for prediction even though the user does not provide all demographic information due to personal privacy. The Bayesian Network is automatically updated when users’ history is accumulated. A single technique is not enough for providing high quality recommendation. The proposed BN strategy is used as part of the hybrid recommendation approach in PPNews.

2.3.4 Mobile Recommendation Systems

Many studies have been undertaken on providing recommendation to users in many different domains such as news (Das et al., 2007; H. J. Lee & Park, 2007), movies (Ono et al., 2009), television (Baudisch & Brueckner, 2005), books (Ziegler, McNee, Konstan, & Lausen, 2005), products (D.-R. Liu & Shih, 2005), activities (C.-Y. Wang et al., 2010), music (J. Lee & Lee, 2007), tourism (Huang & Bian, 2009). As mobile phones are becoming a primary platform for information access,
recommendation technologies have been applied in mobile systems to reduce information overload and provide more focused personalised content for mobile users. However, due to limited input, computation power and interaction capabilities of mobile devices, the development of mobile recommender systems is more challenging. On the other hand, contextual information plays a key role in defining the mobile user’s information needs.

Ono et al. (2009) presented a context-aware movie recommender taking preference and situation (whom, where, mood) contexts into account in the filtering. Ricci and Nguyen (2007) developed a critiques-based mobile recommender that uses location and preference in recommending nearby restaurants. Instead of simple location-based retrieval, users can assign session-based preference such as restaurant with air conditioning to formulate query. Bellotti et al. (2008) offered a mobile activity guide service that takes several user contexts such as location, time, and installed applications into prediction process to recommend nearby activities. Z. Yu et al. (2006) proposed a context-aware media recommender that introduces capability contexts such as device and network capability in addition to common contexts like location. It uses a hybrid recommendation approach that exploits content based approach, Bayesian classifier and rule-based approach. Content-based approach evaluates media items against user preference. Bayesian classifier utilises situation contexts and computes the probability of a media which is suitable for specific environments such as Home/Office, Weekday/Weekend. Rule-based approach filters media items based capability context. Only items which satisfied user devices’ capability such as supported format will be recommended. Su et al. (2010) presented a context-aware music recommendation that combines with different techniques. CF predicts rating with other users who listen to common items in similar contexts environments. CBF computes the pattern of music against user profile. L. Liu, Lecue, Mehandjiev, and Xu (2010) proposed a service recommendation approach that combines semantic contexts similarity and CF based multi-criteria ratings recommendation techniques to predict an overall rating.
The proposed approach also uses contextual information when providing recommendations to mobile users, but distinguishes itself from the above systems for two reasons: First, it focuses on proactive information recommendation while previous work requires users explicit input. Second, previous work is built for a particular domain. JHPeer is a general framework that can support context aware application development in different domains. Recommendation process is executed in numbers of sub-modules. When applying to new application domain, existing modules can be reused and application specific module can be integrated to the system.

2.4 Summary

This chapter has laid out the background into three main research areas developed in this dissertation: mobile information retrieval, context-aware computing and recommender systems. It has discussed how this research is related to other previous researches in context-aware mobile recommender systems.
Chapter 3

JHPeer Framework

Context-awareness is a desired key feature of mobile systems. Context freshness, avoiding single points of failure, proactive information push and fault-tolerance are the key requirements for mobile applications. As discussed in previous chapters, existing frameworks do address some of the key challenges in context-aware applications. However, without a unified solution, these frameworks cannot effectively share context data and support personalised services for mobile users in different application domains.

A holistic approach is needed to address the challenges in context-aware mobile systems, which can incorporate various types of contexts and support context sharing. This chapter presents a hybrid peer to peer framework for efficient organisation, retrieval and management of context data, which enables rapid development of context-aware applications for mobile users. It introduces the three layer model, key components and the design of JHPeer framework. JHPeer is customisable and supports diverse higher level applications with a set of abstractions that are open to many possible implementations.

On the other hand, user profiling is increasingly used to improve user interactions with mobile devices. The limited user input on mobile device means that traditional query based information retrieval is not suitable and proactive systems which supply the user with relevant information without much user input are better suited. User profiling is a way of increasing the relevance of information to mobile users.

The JHPeer framework is extended with a novel user profiling method using Bayesian Network. The BN construction process and the XML schema to store the constructed BN are described. The BN based user profiling method is capable of handling the cold-start problem effectively and can be applied in multiple applications.
3.1 JHPeer Framework

Currently context-aware applications in mobile environment can be developed using low-level APIs supported by modern mobile devices. The developers have to focus too much on the technical aspects and not the services aspects. The key principle of JHPeer is to provide the necessary high-level concepts and APIs for rapid development of various context-aware applications for mobile users.

Figure 3.1 shows the layered structure of JHPeer framework focusing on networking, mechanism and application respectively. This separation of layers not only allows their independent extension, but also simplifies system design by isolating and addressing system properties at appropriate levels of abstraction.

![Figure 3.1 JHPeer Framework](image)

As depicted in the figure, The JHPeer framework is divided into 3 layers.

Network layer: At the lowest level, the layer encapsulates the implementation details such as interconnection networks, network specific APIs. This layer contains various implementations for persistent connection amongst peers and connecting peers via various media.

JHPeer (mobile) engine: In the centre of the framework, this layer provides abstractions of peer and context data management to support various application domains. The JHPeer engine consists of five key components: (1) Context Service
that handles context-related operations such as context discovery, acquisition, analysis, and distribution. (2) Peer Monitor that checks and reports current peer status periodically to nearby peers in order to indicate active mobile devices. (3) Query Service that performs distributed discovery functions for retrieving indexed documents in the peer network. (4) Caching/Indexing that stores, manages, and retrieves documents in local storage and builds index files to enable fast searching. (5) 3rd Party Services that enable developers to implement application specific components such as data replication service, information recommendation services.

The key feature of this layer is reusability and composability. Section 3.3 gives a detailed technical description of the main components in JHPeer engine layer.

Application layer: various applications use the interface provided by the framework’s core functionality for its specific needs. Section 3.4 describes how to build applications on top of JHPeer.

JHPeer framework consists of 2 types of peers – Mobile Peer and Super Peer.

3.1.1 Mobile Peer

Mobile Peers are usually the mobile users with various ranges of mobile devices from mobile phones to mobile PC. A Mobile Peer has a simplified set of functions in order to fit in mobile systems. For instance, news/tourist recommendation system can be deployed in Mobile Peer to receive information from Super Peer and other peers. A Mobile Peer can join any Super Peer that is geographically close to him and act as an edge/ad-hoc peer that allows direct message delivery in the network.

Since mobile devices often have a limited amount of resources such as processing power, battery, memory, display, and network connectivity/bandwidth, etc., these restrict the ability and functionality of mobile devices. For instance, it is not feasible for a Mobile Peer to collect and analyse information from numerous sources as it might overwhelm the resource of mobile device. To resolve this problem, the process which requires high computation should be carried out in Super Peer.
3.1.2 Super Peer

A Super Peer is a full featured node that operates as a service provider server to serve a set of Mobile Peers. For example, a news recommendation service delivers news article to register mobile peers. Multiple Super Peers are connected to each other to provide load balancing and scalability. The difference between a Super Peer and a Mobile Peer is that a Super Peer is responsible for creating and monitoring the overlay network and providing services to other Super Peers and Mobile Peers. Super Peers also provide message propagation and information caching. Mobile Peers can switch from one Super Peer to another whilst maintaining relevant data by using Query service. The details are provided in section 3.3.3.

3.2 Network Layer

Network Layer integrates all possible wired and wireless communication technologies into seamless platforms that enable communication between peers regardless of devices’ hardware and software APIs/protocols. The major function of Network Layer is to establish communication channels and provide reliable messaging facilities. In P2P infrastructure, basic discovery service is also required to locate resources in the peer network such as locating peers.

To achieve this goal, JXTA is selected as network infrastructure in this research. It establishes a virtual network on top of existing networks that hides the underlying physical topology. In the JXTA peer-to-peer infrastructure, peers are organised into peer groups and is communicated via virtual channels which known as pipes. Nevertheless, JXTA provides powerful messaging mechanisms that are used as communication platforms between any devices. But the abstraction layer adds an overhead to normal networking that may not be suitable for time-sensitive applications.

In a heterogeneous mobile environment, mobile devices have different hardware and software. Even though the devices are using the same Bluetooth protocol, the implementation of software APIs are different amongst platforms. Therefore,
abstraction of network layer is necessary to integrate network interfaces. Based on JXTA architecture, adding a network interface is achieved by attaching a new JXTA endpoint messaging module into JXTA endpoint service. For example, the Bluetooth messaging module is implemented specifically for Bluetooth communication.

Furthermore, the network layer deals with the dynamic connectivity in the mobile environment. If one of the network connections is lost, it will try other network interfaces to reach the peer(s) for delivering messages. Besides, the messaging module is responsible for keeping connections alive between peers.

3.2.1 Network Topology

![Figure 3.2 JHPeer Network Topology](image)

The illustration of the connections between mobile peers and super peers is shown in Figure 3.2. The Super Peers are connected in a form of P2P fashion and established the virtual network. Mobile peers connect to one of the Super Peers. The message transmission is propagated inside the peer network. Besides, AD-HOC connection is established dynamically between Mobile Peers via short-range wireless technologies for direct message delivery.
An optional proxy server can be implemented as an external interface to exchange messages between the peers in virtual network and external devices/systems. This benefits supporting more mobile devices such as low-end devices, and increases the flexibility of services such as supporting web services. The communication can be implemented by Short Message Service (SMS) and Wireless Application Protocol (WAP). The implementation of the proxy server is application specific that is not within scope of this thesis.

3.3 JHPeer Engine

JHPeer engine provides abstractions of peer and context data management to support diverse higher level applications, which are open to many possible implementations and enable rapid software development by providing reusable components and reducing the complexity of implementation using low-level APIs. This layer encapsulates the implementation details of context caching, indexing, sharing, searching and coherence protocols, so that the key mechanisms can be used by different applications.

Both Mobile Peer and Super Peer share similar set of components. The context service and caching component are implemented for both mobile and super peers. However, the components in Mobile Peer trend to a service requester or information provider (i.e. contextual information). Additionally, some components are optional in Mobile Peer. For instance, Query service and Peer Monitor components are not indispensable. The caching component is a fundamental component for any applications, which is used to store contextual information and user/application preferences.

3.3.1 Context Service

Context service is designed to handle context-related operations in both Super Peer and Mobile Peer. It is responsible for the representation, gathering, management and supply of contextual information in the peer network. It consists of several sub-components: Context Provider, Context Listener/Context Discovery Listener, Context Filter, Context Protocol, Context Representations and Context Reasoner

34
Figure 3.3 shows the relationship between context service and other components in order to achieve the following main tasks:

- Analysing context information which is obtained from Context providers and notifying Context Listener if contexts changed or is not filtered by associated Context Filter.
- Processing context information via application specific Context Reasoners.
- Pushing and publishing collected context information using Context Protocol in peer network.
- Recording and storing collected context information via Cache/Indexing component (Local Store) or Query Service (Distributed Store).
- Handling and responding to context requests or context queries that are submitted by other service and peers.

In addition, context representations are developed to encode raw data into XML based format in order to manipulate/store/index/transmit context data efficiently in the system.

**Figure 3.3 Relationship between Context Service and Other Components**

The API design of Context Service to support the main tasks is shown in Figure B.1 (in Appendix B). The parameters in the API have not used high-level object (e.g.
Array List). This is because some platforms, such as J2ME, do not include those libraries. Rather than develop different set of APIs for various platforms, JHPeer uses primitive parameters. There are 9 methods related to context acquisition operations: `addProvider()`, `removeProvider()`, `getProviders()`, `asynchronous discoverProviders()`, `synchronous discoverProviders()`, `addContextDiscoveryListener()`, `removeContextDiscoveryListener()`, `addContextListener()`, and `removeContextListener()`.

The `addProvider()` API allows other services or applications to add Context Provider component to Context Service. For example, music recommender system may require noise sensor to determine user environment. This new Context Provider, noise sensor, can be dynamically loaded during system runtime. When a context provider is not required, it is removed by calling `removeProvider()` for saving limited resources of mobile devices. `getProviders()` API returns a list of context providers. The `discoverProviders()` API locate other peers’ associated context providers. There are two types of discoverProviders APIs: Asynchronous `discoverProviders()` just submits requests to target peer(s). The results are returned at anytime and notified by Context Discovery Listener. Hence, it is used with `addContextDiscoveryListener()` and `removeContextDiscoveryListener()` to receive remote providers list and/or remote context events. The synchronous `discoverProviders()` returns result within the given time, otherwise, it returns an error. The `addContextListener()` API registers a components’ context listener and enables the components listen to context events which are generated from context providers. It can be associated with a ContextFilter that filters unnecessary type of context events. If no ContextFilter is associated, all Context Events are forwarded to ContextListener.

The nine other APIs are related to store/retrieve/distribute contextual information that includes save/get/search local contextual documents, and publish/remote search context documents. These APIs re-use other components in the JHPeer engine. For example, the implementation of these APIs use Caching component to store context documents in local store, Query Service to publish, distribute and
retrieve documents in peer network. However, developers can also implement their own policy.

3.3.1.1 Context Provider

Context Provider is a context acquisition module which is used to collect contextual information from physical sensors, such as light, noise, location, temperature, CPU usage, or external sources, such as web services, peer contexts, and smart tags, or logical reasoners. Besides, sensor data is represented using context representation scheme defined in JHPeer. The context data is then sent to Context Service component for further processing.

Any new Context Providers are required to implement the interface class shown in Figure B.2 (in Appendix B). The Context Provider interface consists of 3 methods: 
- `getContextProviderType()` which returns the name of the provider,
- `addContextListener()` which enables context service to receive context event, and
- `removeContextListener()` which removes the existing context services’ context listener when removing Context Provider from Context Service.

A list of context providers defined in JHPeer repository is shown in Table A.1 (in Appendix A). The context providers are not limited to the list. More advanced providers can be integrated into the system.

3.3.1.2 Context Filter and Context Listeners

Context Filter is an interface component, as shown in Figure B.3, which enables developers to implement application specific filtering mechanisms to filter the irrelevant context events before it notifies the Context Listener. It filters contexts in different ways such as filter by types of context document, filter by peers, filter by providers, and filter by the content of context document. For example, a map-based service in mobile peer looks for location context only. Other contexts such as temperature, light will be filtered.

There are two types of Listeners in Context Service: Context Listener and Context Discovery Listener. Context Listener enables other services to receive context
(change) events. The other services only require registering a context listener to **Context Service**. It is optional to register the context listener with a **Context Filter**. If no context filter is associated, all context events received in Context Service will be informed. **Context Discovery Listener** is used to notify two types of remote events: discovery of Context Providers and remote context events including response of context searching and receiving other peers’ contexts. The API of these listeners is shown in Figure B.4.

### 3.3.1.3 Context Representation

A Context Representation is defined in JHPeer in order to store, share, transmit and manipulate various context data in the peer network. There are several ways for modelling context information in variety of context-aware system (Baldauf et al., 2007) such as Key-Value models, Mark-up schema models, Graphical models, Object-oriented models, Logic-based models and Ontology-based models. JHPeer aims to share and reuse context data amongst applications and peers. Simple representation promotes reuse and expandability. A general expandable Context Representation is defined based on object-oriented technique. The context object diagram is shown in Figure 3.4. The IndexableDoc object is on top of the hierarchy. The subclass of IndexableDoc includes ContextAdvertisement, DeviceAdvertisement and UserProfileAdvertisement.

Besides, Application developers can implement their own transform functions, such as toJSON(), toRDF(), to convert context information into the required type that is shown at the end of this section. The default implementation of context marshalling is based on XML which facilitates transmission and interoperability.
IndexableDoc is a subclass of JXTA Advertisement class. It is an abstract document class that is extendable to any relevant document object such as News Article and Product description. The main purpose of IndexableDoc object is to enable sharing, publishing, caching and indexing of context data throughout JHPeer network. The class diagram is shown in Figure B.5. Each document is assigned a unique ID in order to enable indexing and storing of documents in JHPeer engine. The variable adv_lastmodify holds the last modification timestamp, adv_lifetime holds the lifetime timestamp which indicate the length of time for keeping the document in local storage, and the adv_expiration holds the expiration timestamp which indicates the length of time for keeping the document in peers after it is published. For the document which has expired timestamps will be deleted or achieved.

In addition, other data such as persistence (i.e. memory, disk), permission (i.e. private, public, group), multiplicity (i.e. unique, collection, alternative, frequent), and digital signature can be added in actual application implementation depending on software requirements.

ContextAdvertisement holds the dynamic contextual information such as location, weather, device/application status, etc. Each ContextAdvertisement has a Context
ID associated with a Peer ID so it can be published in the peer network. The Context Timestamp attribute indicates when this document is created. ContextModule is an abstract class that can be extended by a specific context object such as Context_CellId, Context_Location, or Context_Address. The XML Schema of ContextAdvertisement and ContextModule are shown in Figure C.1 (in Appendix C).

Context_CellId, Context_Location, Context_Address, and Context_Battery have been developed in JHPeer.

Context_Location holds location information from mobile sensors such as longitude, latitude, altitude, speed, course (i.e. degree to true north), etc. AcquireMethod indicates the acquisition methods. For example, if context data is provided by Java JSR 179 Location API with GPS. The value of AcquireMethod will be “JSR179|MTE_SATELLITE”. If context data is provided by Android’s Network Location Provider, the value will be “ANDROID|Network”. The XML Schema of Context_Location is shown in Figure C.2.

Context_CellId stores the Cell ID information for different types of cellular networks: GSM or CDMA. The Cell ID is used to estimate the approximate user location. The XML Schema of Context_CellId is shown in Figure C.3.

Context_Battery object, as shown in Figure C.4, encloses the dynamic battery status of mobile devices. The DataType variable indicates the type of sensor. E.g. ANDROID, JSR256, NOKIA. The rest of the variables hold the collected raw data.

Context_Address object contains the detailed information of an address. The details of the variables are illustrated in Figure C.5. It is used to provide and share places information in mobile applications. It can also be used to store user’s favourite address in user profile.

DeviceAdvertisement is used to hold static device information such as operating system, memory, battery capacity, hardware/sensors, etc. The details of the variables and attributes are shown in Figure C.6.
**UserProfileAdvertisement** contains user information such as user detail, user calendar, friends, system and application preferences, etc. It holds a set of UserProfileModule that can be extended by specific user preference class such as News Application Profile. The XML Schema of UserProfileAdvertisement and UserProfileModule are shown in Figure C.7.

Developers can add and implement transform functions in order to provide extra functionality such as exporting/providing data to external systems. Below is a simple example. Context_Location object contains 3 variables which hold longitude and latitude data of a User A as follows:

```java
ID contextAdvID = "urn:jxta:uuid-59616261646162614E50472....";
long timestamp = 1315749628000;
double longitude = -1.096508;
double latitude = 50.797716;
```

The default JHPeer implementation will generate a XML that shows below:

```xml
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE jhpeer:Context_Location>
<jhpeer:Context_Location xml:space="default">
  <ContextAdvID>urn:jxta:uuid-59616261646162614E50472....</ContextAdvID>
  <Timestamp>1315749628000</Timestamp>
  <Longitude>-1.096508</Longitude>
  <Latitude>50.797716</Latitude>
</jhpeer:Context_Location>
```

Developers can extend Context_Location class and implement the function "toJSON()" which results in the following JSON format:

```json
{"jhpeer:Context_Location":

```
In order to demonstrate the application and flexibility of the context representation developed in this research, an ontology model proposed in Gu et al. (2005) has been implemented and is illustrated in Figure 3.5. It consists of two levels: Common Context Ontology and Domain-Specific Ontology. The Common Context Ontology contains three fundamental contexts: Context, Device and User. The relationships between these are that Context belongs to a User and/or is generated from a Device of a User. The Domain-Specific Ontology enables designers to develop their own ontology sub-models in different application domains such as News, Movie, Tourism, etc. For example, in news domain, it may require high-level location information (e.g. home, work, current city), news user profile (e.g. news category, average reading time, likely to read at Location/Time).
3.3.1.4 Context Protocol

Context Protocol aims to enable the communication between context services in Mobile Peers and Super Peers. It includes two types of messages: Context Query Message and Context Response Message. Context Query Message is used to transmit the query statement as shown in Figure C.8, while Context Response Message is used to transmit the result of a query and publish context documents as shown in Figure C.9. Both Request variables and Response variables use JXTA StructuredDocument as data type, which can embed XML messaging.

3.3.1.5 Sequence Flow of Context Service

Figure 3.6 shows a sequence diagram of context services in both Mobile Peer and Super Peer.

In Mobile Peer, Context Providers are created in the application and registered to Context Service. Other Components such as Tourism service will register a Context
Listener to Context Service. When Context Provider notifies a context event to Context Service, Context Service will start to process. It may publish the context to Super Peer. However, publication of contexts should only happen periodically or when it is needed in order to save battery life. At the same time, it will notify the registered Context Listeners, if it is not filtered. The application can adapt its behaviour based on notified contexts. The context is cached by Caching component (Local) or Query Service (Remote) for future usages, e.g. when performing content based filtering and/or collaborative filtering. Beside, Context Service listens to the Context Query and provides response.

In Super Peer, the context processing is similar to the one in Mobile Peer. However, it may consist of an additional component: Context Reasoner. Context Reasoner is used to analyse the low-level context (e.g. GPS location) and infer to a high-level context (e.g. Home, work). It is usually implemented by the application services since different services may require a variety of high-level contexts. Another difference of Context Service in Super Peer is that it propagates Context Query to other peers and collects results for Mobile Peer.
3.3.2 Caching/Indexing Components

Caching/Indexing Component is responsible for storing/indexing IndexableDoc object in local storage (e.g. disk, flash drive, memory) and building index files to enable fast searching. It is implemented in both Mobile and Super Peer with same set of API shown in Figure B.6 (in Appendix B). A large volume of stored context information may overwhelm a mobile device; therefore an expiry timestamp is assigned to each IndexableDoc. Caching/Indexing component will clean up the expired IndexableDoc from the local storage.

Furthermore, the cache API can be re-used in implementing multiple cache components with different implementations in system runtimes. For example, Context Service may require implementing a distributed cache system (mirroring)
to hold peer context documents. A Forum service may implement a centralised or distributed SQL-based cache system. These caching systems are implemented based on cache API in JHPeer framework.

3.3.3 Query Service

Query Service is an advanced discovery service that includes four major functions: performing local or remote searching, publishing IndexableDoc document in peer network, propagating query to nearby peers, handling advanced query such as Boolean Query, XQuery, SQL-Liked Query, etc. The Query Service API is shown in Figure B.7. The reason for implementing query service is because JXTA discovery service is only designed for publishing advertisement and performing plain text searching. It can only handle simple keywords query which is insufficient for modern applications. Current mobile devices are capable of storing more information and are becoming a significant information source. Query Service supports full text searching on Super Peers and Mobile Peers. Since it extends JXTA discovery service, it also incorporates JXTA Distributed Hash Table (DHT) index (Traversat, Abdelaziz, & Pouyoul, 2003) (in Super Peer only) for fast searching in distributed peers.

In normal operation a Mobile Peer joins a Super Peer that is geographically close to it. When a Mobile Peer switches from one Super Peer to another, the new Super Peer will get the user’s information from previous Super Peer by calling Query Service. When a Mobile Peer is disconnected, updated information will be published to the peer network.

Query Protocol extends JXTA discovery protocol. The Query Request Message, as shown in Figure C.10, supports multiple attributes/values. The Query Response Message, as shown in Figure C.11, encodes the results in a JXTA Structured Document, which is not limited to string data.
3.3.4 Peer Monitor

Peer Monitor is designed to check online peer status periodically. This component is essential because Mobile Peer may frequently disconnect from the peer network in mobile environment. Since JXTA protocol does not provide mechanism to keep the real-time online status of peers, a Peer Monitor Protocol is developed to query and return the type of peer, associated services (e.g. news/movie service), peer information (e.g. uptime, ping speed) and state of a peer. The state of peer is represented in 6 levels: UNKNOWN, ONLINE, AWAY, BUSY, INVISIBLE, and OFFLINE, which can be assigned by relevant applications automatically or by the user manually. Peer Monitor class is shown in Figure B.8. Recommendation services are based on a user’s status. For example, if a user sets his/her status to BUSY, the proactive recommendation services will postpone information pushing to the user. In order to minimise energy consumption, Peer Monitor in Mobile Peer will only submit information to Super Peer when a peer’s status changes. Super Peers only check Mobile Peer’s status after an inactive period. When the battery energy is low, Peer Monitor automatically increases the checking time interval.

Peer Monitor also provides access to real-time peer information by monitoring several JXTA network components such as Rendezvous Service for connected Peers, Discovery Service for Peer Information and PeerInfoService (in Super Peer only) for service monitor report.

Similar to other JHPeer components, PeerMonitor also provides remote querying functions such as getRemotePeerStatus(). The returned results will be notified by PeerMonitorListener.

Peer Monitor Protocol is designed to check the status of peer, and check/keep connection alive. Figure C.12 and Figure C.13 (in Appendix C) shows the design of Peer Monitor Query Message and Peer Monitor Response Message respectively. The Request and Response is designed to hold StructuredDocument. StructuredDocument may generate overhead for time-sensitive query (i.e. ping).
ExtraData variable is used as an alternative for performing these types of operations.

### 3.3.5 3rd Party Services

3\textsuperscript{rd} Party Services can be integrated into JHPeer for providing functionality to users or other system components.

The implementation of 3\textsuperscript{rd} Party Services only requires extending the service interface class: com.jhpeer.service.Service. It then registers the 3\textsuperscript{rd} party service instance to com.jhpeer.service.ServiceManager for receiving system status such as startApp(), pauseApp(), or stopApp().

### 3.4 Application Layer

Application layer allows a wide spectrum of mobile applications to be developed on top of JHPeer engine layer.

In Super Peer, application layer is mainly used to deploy as user interface for system administrators to control the application. Implementation is not limited to stand alone user interface and can be a web interface. The applications can access the provided JHPeer APIs.

In Mobile Peer, user interacts with the system via user interface. In modern mobile platforms, JHPeer engine normally runs in background. Figure 3.7 shows multiple applications accessing JHPeer engine in modern mobile platform.
3.4.1 Application Scenarios

The following application scenarios are an indication of how the system could be used:

**Mobile News**: Imagining that you just arrived in a city, without formulating any queries, your mobile news recommender automatically reports local traffic information, or latest local news based on your preference and contexts. All such information will display on your mobile device. The biggest benefit of this service is that the user could make timely decisions based on most updated news or information. For example, a bus strike happens. The service will alert the user and the user can be aware of his travel conditions and plan accordingly.

**Mobile Discussion Board**: Discussion board may be applied in different areas, e.g. communicating the quality of one product or the situation of an on-going event (e.g. football game). Mobile discussion board provides an information sharing platform to enable users to express their feeling or to discuss any items (i.e. products, events, shops). A good example is the university discussion forum. Since lecturers and
students constantly post information about the subject, students can receive information that is of interest to them as fast as possible, rather than browsing the forum. For instance, if a lecture was cancelled, students would definitely want to know about it in advance. By supporting proactive information push, users can be notified of any relevant posts in the forum.

**Tourism:** Tourism services could enhance the experiences of tourists, as well as locals. Once a user starts the application, system provides local information depending on his or her geographical position. For example, a mobile user is located in London. The application acknowledges the location where the user is via GPS/AGPS and offers some useful recommendations, e.g. address of the nearest Hotel, special event at the popular pub or sales information at the nearest shopping centre. If the user selected the Ibis (Hotel) entry, the system will provide further personalised information such as the directions to the hotel and the price of services (i.e. travel expenses).

### 3.5 User Profiling

User profiling is a key technology in personalised applications. User Profiling attempts to estimate the importance of information to a user at a particular time and space so that information can be recommended to them by a recommender system. A user profile is a collection of personal information that describes the interests of an individual user, which is generated by monitoring the actions of the user in explicit or implicit ways during application usage.

Typically, users are interested in information which is around their physical location, such as shops, restaurants or public transportation. Tracking the user's current location along with the locations the user has visited previously is useful in generating a profile of location interests. Monitoring user activities such as web searching is suggested for capturing non-local and longer lasting user interests. This sort of activities can take place over many weeks or longer, generating vast amounts of search queries and downloaded files. It is possible for an application to analyse this information and identify certain topics that a user is interested in and
use this in a recommendation system. The advantage in profiling users based on their activities is that data capture is implicit and does not require any extra effort from the user.

However, typical recommender systems have to face data sparsity and the cold-start problem, where no information is available on the first use of a recommender system. A Bayesian Network Method has been developed to construct user profile in JHPeer network.

3.5.1 Bayesian Network in User Profiling

A Bayesian Network is a probabilistic graphical model that combines the advantage of CF and CBF. It means, for a new user, the system will use group profile data and generate a Global BN that can not only solve the data sparsity problem, but can also provide a variety of options. After the user uses the application for a while, the system will revert to using the contents of the user’s profile to make recommendations that are more suitable for that particular user. Additionally, BN could provide real-time prediction to achieve the objective of just-in-time personalisation because BN requires a smaller memory and provides faster computation than other CF/CBF techniques (Breese et al., 1998).

In a recommendation system, Bayesian Network (BN) can be used to estimate a user’s interest and predict an uncertain user’s situation based on the user or group’s previous activities and history. To accomplish this objective, three phases are designed to fulfil the requirement of a Bayesian Network which are construction, prediction and revision phase.

3.5.1.1 Construction of BN

The Bayesian Network construction follows three steps: selecting relevant variables (nodes), specifying Bayesian Network, and defining conditional probability tables.

It classified four types of relevant variables which can be applied in recommendation domain: demographic variable, situational-behaviour variable, psychographic variable and domain specific variable. The values of variables are
based on common Boolean values (e.g. T/F, YES/NO), ordered values (e.g. LOW/MEDIUM/HIGH), and integral values (e.g. 1-20, 21-40, 41-60).

The demographic variable contains gender, age range, occupation status, education, income, religion, nationality, etc. These attributes can influence the user’s interests to a large extent. For instance, in news domain, men are more interested in sports news than women; in movie domain, more women prefer romantic movies than men. As the users are getting older, the interests of users may change such as clothing, activities, and reading materials.

The situational-behaviour variable is related to the user status and user ambient contexts in particular situation that includes location, usage period, time spend, weather, mood with whom etc. For instance, in movie domain, if a user is dating a girl, the chance of choosing a romantic movie is greater than choosing an action movie. In News domain, it is possible to determine user behaviours. If a user usually reads news in the morning from Monday to Friday, the probability that the user may want to read news during weekdays is higher than weekends. The proactive recommender may push news at that particular time.

The psychographic variable is considered as personality, lifestyle, characteristics, values, attitudes, hobbies, and general interests. Personality, lifestyle and characteristics can determine how a person treat or react to a situation. These factors reflect user’s behaviours and affect the user needs for certain recommendations. For instance, if a user’s personality trait is Openness to Experience/Intellect and extroversion, the user may like to try new things. The tourism recommender may recommend new types of places that the user may not have tried before such as a newly open restaurant and new type of cuisine.

The domain specific variable is comprised of different attributes in each domain. For example, in Movie domain, a developer needs to consider movie genre, actor etc., whilst in tourism domain, the possible attributes are tour motivation, cost and activity types.
After the relevant variables are selected, there are two approaches that can be used to specify the BN structure. One is to join variables manually. The other is to learn the BN structure automatically via history data.

Finally, the conditional probability table (CPT) of each variable has to be defined. There are several techniques available to developers. Firstly, the CPT can be assigned manually. Secondly, it can be carried out by a questionnaire survey. Thirdly, it can be built using sampling algorithms based on collected data.

### 3.5.1.2 Prediction (BN Influence)

The major task of this phase is to compute the posterior probability distribution of a target node by using any given evidence nodes. The computation is based on a chain rule of probability theory which defines an equation (3.1) where $x_1$ to $x_n$ is the order of nodes in BN, $x_i$ is the target node, and $\text{Parents}(X_i)$ is the set of parent nodes of $x_i$. A specific posterior probability value is used to estimate the user’s interest.

$$P(x_1, x_2, ..., x_n) = \prod_{i=1}^{n} P(x_i | \text{Parents}(X_i))$$

(3.1)

In addition, soft evidence is also employed to improve the prediction instead of using a user’s Local BN (i.e. Local BN is generated for particular user based on his historical data). It can be computed using user history. For instance, in News domain a BN network is created for predicting user interested news categories based on demographic variables (i.e. age, gender). When user history is accumulated, the system computes the distribution of user interested categories such as 40% on Headlines, 29% on Nation (UK), 5% on Sports, 12% for entertainment, and 14% on technology. These values reflect the user interested news categories and are used as soft evidence in the BN to improve prediction (Detail example is illustrated in section 5.5.1.2).
3.5.1.3 Revision of BN (Offline Process)

Revision of BN is divided into two parts: revision of Global BN and revision of Local BN. When a user is new to the system, Global BN is applied. Whilst the user’s history is accumulated, the user’s Local BN is being generated. Both BNs are revised based on incremental learning (Korb & Nicholson, 2004). Besides, incomplete data can be resolved by Gibbs sampling or Expectation maximisation (EM) algorithm (Korb & Nicholson, 2004). Therefore, the system can effectively eliminate the problem of data sparsity for new users.

In addition, the revision process is carried out as an offline process that may be scheduled on an hourly or daily basis, since updating BN may cost a lot of computation power.

Furthermore, Bayesian Network can be extended in this phase that additional variables or sub-BNs can be added to the BN.

3.5.2 Bayesian Network Advertisement

In order to store and distribute the constructed BNs in JHPeer network, a XML schema, BayesianNetworkAdvertisement, is defined and used as a part of User Profile (i.e. User Profile Module). The BayesianNetworkAdvertisement holds the most essential values of a BN such as node, parents of node, Conditional Probabilities Table of nodes, explanation phrase and so on. It can be converted into common BN file formats such as NET, DNE, and XMLBIF. Figure 3.8 shows the XML Schema of Bayesian Network Advertisement for JHPeer Framework.
Figure 3.8 XML Schema of Bayesian Network Advertisement

The following shows the example XML document of tourist Bayesian Network. The XML document enables the distribution of BNs in JHPeer network. The BN APIs which provided in JHPeer framework can save the BN data into other file format for interoperability.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE jhpeerapp:BayesianNetworkAdvertisement>
<jhpeerapp:BayesianNetworkAdvertisement xmlns:JHPeer="http://JHPeer.org">
  <BayesianNetworkID>urn:jxta:uuid-59616261646162614E50472050325033476C6F62616C4E65B773824E10</BayesianNetworkID>
  <NetworkRadius>18.0</NetworkRadius>
  <NetworkName>Tourist_BN</NetworkName>
  <TreeNode>
    <NodeType>0</NodeType>
    <Name>Gender</Name>
    <description>Gender</description>
    <State>Male</State>
    <State>Female</State>
    <InformationType>3</InformationType>
    <Probability>0.482|0.518</Probability>
  </TreeNode>
  <TreeNode>
    <NodeType>0</NodeType>
    <Name>Age</Name>
    <description>Age</description>
    <State>Youth</State>
    <State>Adult</State>
  </TreeNode>
</jhpeerapp:BayesianNetworkAdvertisement>
```
3.6 Summary

This chapter has described the JHPeer framework based on JXTA and hybrid P2P technology, which supports the acquisition, representation, storing, indexing, sharing and delivery of context information and provide modular components that are common across applications. The usage of JHPeer framework for context-aware application can ease and dramatically accelerate development of specific context-aware applications.

JHPeer overcomes the key challenge upon the heterogeneous mobile environment, which is to integrate these technologies into a computational framework seamlessly and enable communication between each other. It enables the
distribution of context information in peer network to handle the rapid growing number of mobile users seeking dynamic information. Third party components can be added to extend the functionality. The components are reusable in context-aware applications. Designers only need to focus on application-specific tasks. JHPeer assists and eases the development of context-aware applications for mobile users. Chapter 5 demonstrates how JHPeer framework is used to build a mobile news recommender system.

An BN based method is developed to generate user profile and estimate user’s interest and predict an uncertain user’s situation depended on their previous activates and history. A Global BN is generated by group profile data for predict a new user’s interest, which overcomes cold-start and data sparsity issue, the typical recommender problems. Additionally, the proposed BN approach enables to offer real-time prediction service that achieves the objective of just-in-time personalisation due to its smaller memory requirement and faster computation than other CF/CBF techniques. One of the most challenging problems in using a Bayesian Network is the model construction. This thesis develops a model construction process with many variables for context-aware mobile environment. The possible types of variables have been identified that is suitable for use in recommendation systems. The effectiveness of the user profiling method is also demonstrated through experiments, which is described in Chapter 6.
Chapter 4

Analytic Hierarchy Process based Multi-Criteria Ranking Approach

Most of the current recommendation systems deal with single criterion ratings. This chapter presents an analytic hierarchy process (AHP) based multi-criteria ranking approach called AHP-MCR approach. AHP-MCR takes user contexts into account and rate relevant information for personalised services. Four common criteria, which can be used in context-aware recommender systems, are identified and selected by investigating a wide range of literature.

4.1 Motivation

The ubiquitous mobile devices are embedded with more and more sensors and relatively large storage, contextual information is becoming richer in mobile environments. It has added great benefits when applied to mobile devices as it maximises the effectiveness of the limited screen size and processing power. By using contextual information, a mobile recommender has the potential to deliver more relevant and just-in-time information to mobile users with reduced keystrokes.

Unlike traditional recommenders which are only based on users and items space, a context-aware recommender system needs at least three spaces: users; items; contexts. In order to handle contextual criteria and meet the diverse requirements of different applications, a rating approach must be able to incorporate multiple criteria and adapt to different application domains.

4.2 AHP-MCR Approach

The AHP-MCR approach is developed to deal with multi-criteria problems in context-aware recommender systems. While most of the traditional recommenders only calculate a single rating for one item, AHP-MCR has the advantage that
considers more than one criterion and provides more possible combinations in order to make a recommendation. For instance, if there are four criteria in the model, the system may only score Criterion 1 and Criterion 3 for Situation 1, and score Criterion 2 and Criterion 4 for Situation 2. In this case, the approach can offer recommendations of top-k for one criterion or combination of several criteria. Compared with many single-criteria recommendation approaches, the AHP-MCR approach has the following advantages:

1. It organises the recommendation process into a hierarchy that is flexible to embed dynamic context criteria in the criteria hierarchy.
2. It allows estimating items rating by combining several CF and CBF algorithms. Different algorithms can be used in different criterion. For example, CBF algorithms can be used in User Profile Criteria to predict user interested items and CF can be used in User Situation Criteria to predict user interest in current environment through similarity in community histories.
3. The multiple ratings of the criteria can be used as an indication of user needs for the purpose of proactive information pushing.
4. User customisation on criteria importance can take place in the ranking process that increases the functionality and flexibility of the recommender. For example, in news domain, user normally prefers to read popular stories, but he may change his mind when events happen (e.g. football match, natural disaster). At this moment, he can adjust the criteria importance to allow a proactive news recommender to push first-hand related news as soon as possible.

AHP-MCR derives relative weights of criteria from pairwise comparisons by the importance of each criterion. There are two common methods in deriving weights: eigenvector method and geometric mean method. However, the eigenvector method suffers from the problem of rank reversal (Barzilai, 1997). The AHP-MCR approach is based on geometric mean method. The procedure for recommender is summarised as follows:

2. Construction of criteria comparison pairwise matrix.
3. Derivation of weight of criteria from comparison pairwise matrix.
4. Computation of rank for each alternative.

4.2.1 Construction of Criteria Hierarchy and the Alternatives

This stage is to identify criteria hierarchy and locate the recommendation items. The criteria can be fixed structure or can be selected based on the availability of user contexts. The user contexts can be a user preference or existed contextual information. For instance, in News domains, user can submit his requirement of news such as highly rated news, nearby news. The news items which met these criteria will be assigned a higher score. However, if user location context is missing, the nearby news criteria will not be selected. The selection of alternatives is an information retrieval process, but it varies in different applications/domains. The information can also be retrieved based on user preference and existing user contexts. For instance, in News domains, the news articles can be limited to particular sources, periods, user keywords (query), and location/region.

A general criteria hierarchy for mobile recommender is developed and is shown in Figure 4.1.

![General Recommendation Criteria Hierarchy](image)

Figure 4.1 General Recommendation Criteria Hierarchy

These criteria are selected by investigating the common features of recommendation systems in various domains from existed research works. Four
general criteria are identified and defined: Profile, Situation, Rates and Attributes. These four criteria are suitable for different domains such as News, Movie, and Tourist, etc. The profile criterion can be further divided by four types of variables, which is described in section 3.5.1. In addition, nested hierarchy is supported in the construction for flexibility.

**Profile Criterion** represents a set of user profile variables to predict the priority of items which may be of interest to the user. The score of Profile Criterion is computed based on the user’s preference and user’s history. It can also be referred to as $R: \text{user(s)} \times \text{items} \rightarrow \text{ratings}$ in a traditional recommendation system. The sub-criteria of a profile can be selected based on the four types of identified variables: demographic variables, situational-behavior variables, psychographic variables, and domain specific variables.

The demographic variables are a typical value to group people into different social clusters such as age groups and economic groups. The demographic data can influence the user choice in different domains. For example, in news domain, an economist may prefer reading business, finance, and politics news.

The situational-behaviour variables refer to user preference and behaviours. The typical example of this variable is the preferred time and location which is computed by the user histories, for example, a user’s frequency of reading news at office in the morning. This can be used as an influence factor for proactive recommenders.

The psychographic variables are the general value of user’s personality, lifestyle, values, attitudes, hobbies, and general interests. This factor can reflect the user characteristics and behaviours. However, not much recommendation systems require users to provide this information. The recommendation systems are usually based on domain specific variables to compute user interest.

Domain specific variables are the user interests in particular domains. For instance, in news domain, it reflects the news category, keywords or topics, etc. In scholar
recommendation, it reflects keywords, subjects such as information retrieval, context-awareness, collaborative filtering and information systems.

**Situation Criterion** handles various temporal contexts variables that are relevant to current environment. Sub-criteria can include Emotion, Status of User, Weather, Location, task, with/nearby friends, etc. The score of this criterion can be calculated based on CF algorithms (e.g. reduction-based CF (Adomavicius et al., 2005)) or other rating-based algorithms. For example, in tourism domain, a recommender tries to deliver interesting Point-Of-Interests (POIs) to user. However, if the current weather indicates heavy rain, the indoor POIs should receive a higher score.

**Rates Criterion** is used to measure an item’s popularity. Higher click rate or positive votes of item means that the item will get a higher score.

**Attributes Criterion** refers to an item’s attributes in a particular domain. This criterion is domain specific. For example, in news domain, it may be divided into two sub-criteria: Time and Site Subscription. Time is used to compute the score of items based on recency such that a recent news article has more weight than old news and the weight is reduced as time passes. The score of Site Subscription criterion is computed based on the number of subscription to a news web site. For movie domain, Director, Writer, Actors and Award may be used to compute the score based on their ratings.

### 4.2.2 Construction of Criteria Comparison Pairwise Matrix

In the literature, the comparison pairwise matrix (CPM) \((p_{ij})\) for each criterion pair \(i\) and \(j\) is equal to \(p_{ij} = C_i/C_j\), where \(C_i\) and \(C_j\) are the user assigned numerical judgment (value: 1-9) (Saaty, 1980) of Criterion \(i\) and \(j\) respectively. By adopting AHP-group decision making, the assignment of criterion pair’s importance can be generally classified into three different approaches: individual-based, group-based and mixed approach.

With individual-based approach, the importance of criteria is judged explicitly by the user or computed automatically using user history.
Group-based approach adopts the techniques in AHP-group decision making called aggregation of individual judgments (AIJ) (Forman & Peniwati, 1998). The importance of criteria is aggregated from users. There are three possible methods:

- **Method 1:** By aggregating all users’ numerical judgment (assigned or/and computed judgment) and computed by arithmetic mean for each criterion.

- **Method 2:** By aggregating numerical judgment (assigned or/and computed judgment) from users that have similar contexts and computed by arithmetic mean. It can apply common similarity methods such as Pearson correlation coefficient, Cosine similarity, Euclidean distance, k-nearest neighbours, clustering (e.g. k-means clustering), etc. to find users.

- **Method 3:** By applying machine learning techniques to calculate likelihood values such as Bayesian prioritisation procedure (Altuzarra, Moreno-Jiménez, & Salvador, 2007) or fuzzy preference (Ekel, Queiroz, Parreiras, & Palhares, 2009; Mikhailov, 2004) on incomplete data sets.

Mixed Approach enables the combination of both individual-based and group-based approaches allowing assigned personal importance and global users’ importance to be taken into account. The final importance is computed by arithmetic mean. For example: a set of numerical value of criteria \( i \) is defined as \( X_n \), where \( n \in N, N = \{1,2,3\} \), \( n = 1 \) means it is user assigned individual numerical value, \( n = 2 \) means it is normalised numerical value which is computed from user history, \( n = 3 \) means it is the group numerical value between 0 and 10. The importance of criteria \( i \) is the arithmetic mean of \( X_n \) that is represented as \( C_i = \frac{1}{n} X_n \).

### 4.2.3 Derivation of Weight of Criteria from Comparison Pairwise Matrix

Once the comparisons matrices are specified, the weight of criteria \((w_i)\) is computed using the geometric mean as follows:

\[
w_i = \left( \prod_{j=1}^{n} p_{ij} \right)^{1/n}
\]
where \( n \) is the number of criteria in the same level of hierarchy and \( p_{ij} \) is the numerical value from section 4.2.2.

### 4.2.4 Computation of Priority for Each Alternative

The priority of each alternative requires normalisation before the computation of the final score. The alternative (item) normalised priority \( (N_{ij}) \) is calculated as follows:

\[
N_{ij} = \frac{n_{ij}}{\max_k r_k} \tag{4.2}
\]

where \( r_{ij} \) is the assigned priority of criteria \( (p_{ij}) \) by any rating-based algorithms and is divided by the maximum value of the priority \( (r_k) \) in corresponding criteria \( (p_{ij}) \).

The last procedure is to synthesise all scores across all criteria in each hierarchy-level from bottom to top to determine the final priority. The score of an item \( (score_i) \) is computed as follows:

\[
score_i = \sum_{j=1}^{n} N_{ij} w_j \tag{4.3}
\]

where \( N_{ij} \) is the normalised score of criteria from equation (4.3) and \( w_j \) is the weight of criteria from equation (4.1).

Only top-\(k\) alternatives from the ranked list will be pushed to the users’ mobile devices based on recommendation approach.

### 4.3 Applications of AHP-MCR Approach

The following examples show how the AHP-MCR approach can be applied in different recommendation domains. For simplicity, the examples only contain 2 level of criteria hierarchy.
4.3.1 Movie Domain

Mobile movie recommender aims to provide just-in-time recommendation of movies in nearby cinema. The design of ranking factors can fit into the four major criteria (Table 4.1): Profile, Situation, Rates and Movie Attributes.

For Profile Criterion, it may be made up with two sub-criteria: User Interest and User Behaviour. User Interest predicts the ratings of movie types using user variables such as age, gender, and personality. CF techniques (i.e. correlation coefficient) or the proposed BN strategy can be used in this computation. User Behaviour predicts the ratings of each movie based on user history via CBF techniques. The content of previous movie can be an indication of user interest. For instance, previous history shows that a user is interested in watching action movie directed by Director A. A new action movie directed by the same director will get a high score.

For Situation Criterion, it represents a set of user situation and context attributes such as cinema nearby or near to a particular location, show times, seats left at cinema. These dynamic attributes may influence user decision. For example, if user is going to watch with his family, a film which is suitable for family (i.e. film classification: U, PG, 12A) should be highly scored.

For Rates Criterion, it represents a set of users’ ratings or votes. The ratings can be a simple value that scales from 1 to 10 or can be a set of sub-ratings such as Story, Acting, Direction, and Visuals. It can also rate the quality of cinema or sub-rating such as cleanness, comfortable, screen quality, and service. The votes can refer to the number of users that would recommend the film to friends.

For Movie Attributes Criterion, it represents set of movie domain attributes such as movie popularity, famous director/writer/actor, country, awards, gross revenue, and release date (recency). A higher score will be assigned for more famous or more recent films.
Table 4.1 Example of Movie Domain Criteria Hierarchy

<table>
<thead>
<tr>
<th>Criteria</th>
<th>Sub-Criteria</th>
<th>Variables</th>
</tr>
</thead>
<tbody>
<tr>
<td>Profile</td>
<td>User Interest</td>
<td>age, gender, personality</td>
</tr>
<tr>
<td></td>
<td>User Behaviour</td>
<td>previous watched/rated movie, preferred genre/directors/actors</td>
</tr>
<tr>
<td>Situation</td>
<td>Location</td>
<td>longitude and latitude, address</td>
</tr>
<tr>
<td></td>
<td>Time</td>
<td>current time/preferred time</td>
</tr>
<tr>
<td></td>
<td>Whom</td>
<td>whom</td>
</tr>
<tr>
<td></td>
<td>Availability</td>
<td>sit left in cinema</td>
</tr>
<tr>
<td>Rates</td>
<td>Ratings</td>
<td>ratings of film, ratings of cinema</td>
</tr>
<tr>
<td></td>
<td>Votes</td>
<td>votes of film, votes of cinema</td>
</tr>
<tr>
<td>Movie</td>
<td>Popularity</td>
<td>famous director/writer/actor, country, awards, gross revenue</td>
</tr>
<tr>
<td>Attributes</td>
<td>Recency</td>
<td>release date</td>
</tr>
</tbody>
</table>

4.3.2 Tourism Domain

There are various types of tourism recommenders. One of the popular mobile tourism recommender is to recommend point of interests (POIs) (places) such as attractions, viewpoints, shopping centre, hotel, and so on. The possible design of criteria hierarchy is shown in Table 4.2.

For Profile Criterion, User Interest sub-criterion can predict the user’s place of interest category, types, and budget range based on demographic and psychographic variables such as resident of city/country, age range, occupation, social economic grade, marital status, kids, and personality. User Behaviour can
estimate the user preferred visiting time/visiting area/distance/length of tour by using user history or preference.

For Situation Criterion, it is used to rank the POIs to fit user ambient environment. The considerable sub-criteria include location/region (distance), time (opening time), whom, and weather. The algorithms that apply in these criteria are various. For instance, distance can apply graphical distance measurement, and weather can apply defined score based on the condition of place (i.e. indoor/outdoor).

For Rates Criterion, it can include rating, votes that reflect user satisfaction on different factors. The place with higher ratings will be assigned a higher score.

For Tourism Attributes Criterion, the possible sub-criteria include popularity of POI which can be estimated by statistical data (e.g. number of people visiting) or the number of photos which were tagged for a particular location from web photo albums. More recent digital cameras are embedded with GPS. Photos taken by these cameras usually contain location metadata and can be published on web albums. It can be used as an indication of POI popularity. Besides, other variables such as awards for POI (i.e. shop) and official suggestions can also be considered as the popularity of POI.

Table 4.2 Example of Tourism Domain Criteria Hierarchy

<table>
<thead>
<tr>
<th>Criteria</th>
<th>Sub-Criteria</th>
<th>Variables</th>
</tr>
</thead>
<tbody>
<tr>
<td>Profile</td>
<td>User Interest</td>
<td>resident of city/country, age range, occupation, social economic grade, marital status, kids, personality</td>
</tr>
<tr>
<td></td>
<td>User Behaviour</td>
<td>user history of visiting time, visiting area, distance, length of tour</td>
</tr>
<tr>
<td>Situation</td>
<td>Location</td>
<td>longitude and latitude, distance to POI</td>
</tr>
<tr>
<td></td>
<td>Time</td>
<td>opening time and current time</td>
</tr>
<tr>
<td>Rates</td>
<td>Ratings</td>
<td>ratings of POI</td>
</tr>
<tr>
<td>--------------</td>
<td>--------------</td>
<td>----------------</td>
</tr>
<tr>
<td>Votes</td>
<td>votes of POI</td>
<td></td>
</tr>
<tr>
<td>Tourism Attributes</td>
<td>Popularity</td>
<td>number of people visiting, number of tagged photos on web album, awards, official suggestions</td>
</tr>
</tbody>
</table>

### 4.4 Summary

This chapter developed an AHP-based Multi-Criteria Ranking approach to generate recommendation for context-aware recommender systems. The proposed approach handles complex multi-criteria problems by using AHP criteria hierarchy. A general criteria hierarchy model is developed through empirical studies. The criteria hierarchy is extendable by adding sub-hierarchies, which enable to embed more contextual information provided by additional mobile sensors and third party sources (i.e. general user interest from news domain, user travel habit from tourism domain). The weights of the contexts (criteria) can be assigned by user or automatically adjusted via individual-based and/or group-based (group decision making) assignment. Any rating-based algorithms can be used for scoring criteria that include CF and CBF. Finally, several criteria hierarchy examples of different recommendation domains are provided as a reference to develop context-aware recommender using the AHP-MCP approach.
Chapter 5
Design and Implementation of a Proactive Context-Aware News Recommender

To evaluate JHPeer and AHP-MCR approach, a news recommendation system has been developed: Proactive Personalised News recommender (PPNews), in context-aware mobile environment. The PPNews recommendation system is built upon the JHPeer framework. The AHP-MCR approach is deployed to recommend personalised news to mobile users. BN-based user profiling technique is used to build user profile and handle cold-start problems. This chapter describes the design and implementation of PPNews.

5.1 Introduction

PPNews provides personalised news services to meet mobile users’ needs (Yeung & Yang, 2010). The key features include: proactive, context-aware and personalised. It is built on top of JHPeer framework that incorporates user contexts and news article attributes for automatically pushing real time news which are considered relevant to mobile users. The delivery of news item is faster than using traditional push-pull methods.

PPNews application is implemented using Java programming language for both Mobile and Super Peer. Since it is based on pure Java, it can ideally run on any Java-enabled mobile devices, including Android mobile platform.

5.2 System Overview

Figure 5.1 illustrates the software architecture of the prototype implementation.
Similar to the design architecture in Figure 3.1, PPNews consists two types of peers: Mobile Peer and Super Peer. A Mobile Peer connects to one of the Super Peer and submits a user’s profile. The context service in Mobile Peer will automatically push user changing contexts to Super Peer. The Super Peer constantly collects real time news articles from news contents providers via Really Simple Syndication (RSS). Instead of asking users to input a query directly from their mobile device, the Super Peer filters and ranks new news articles based on a user’s profile, location, usage patterns, peer ratings and news article attributes. A user can change the weights of each ranking component depending on personal interest at any time. Top-k (The value of k can be customised by user) news stories of each category will then be proactively delivered to the user’s mobile device based on user’s status. During the interactions with the user, the system collects the user’s reading patterns, location, peer activities and use these contextual information to predict the user’s interests and update the user’s profile.
5.3 Implementation of JHPeer Components

This section describes the implementation of JHPeer Components for PPNews application in detail including the libraries used. Since different applications may have various requirements, the implementation of JHPeer Components may vary between applications. As discussed in Chapter 3, the JHPeer is a general framework that supports context-aware applications in mobile settings. JHPeer provides a set of APIs and basic implementation of the fundamental components. In order to enable a longer battery lifespan for mobile devices, complex computation is usually handled by Super Peer. Mobile Peer is mainly used to capture user low-level contexts (sensor data, raw data) and user operations, while Super Peer is responsible for processing the user contexts and providing service to Mobile Peers.

5.3.1 Implementation of Context Service

The Context Service in Mobile Peer is implemented to push low-level contexts to Super Peer periodically in the form of ContextAdvertisement. The implementation of Context Service supports filtering and provides (particular) context events for other services through ContextListener, where the context events are generated by Context Providers. The Context Providers in PPNews are capturing the context data from GPS/accelerometer sensors, CellID (location), battery sensor and device’s calendar (current meeting or not).

In order to maximise battery, some context providers such as GPS provider will be stopped when receiving a pause event. For example, application will move to the background or screen is turned off.

The Context Service in Super Peer is implemented to receive and aggregate contexts from mobile peers. In current implementation, a simple context reasoning is implemented to find relevant user location (address) and provide weather information by using Yahoo APIs\(^1\). Since high-level contexts such as a “user is

\(^1\) http://developer.yahoo.com/
5.3.2 **Caching Service**

The Cache Service is implemented using different libraries in PPNews. In Super Peer, Apache Lucene (2011) library was implemented for providing fast index and search facilities. Term Frequency (TF) and Inverse Document Frequency (IDF) is calculated and included in the inverted file. News items can be retrieved from the local storage quickly with the additional information associated with the news item like TF and IDF, which benefit the computation of recommendations.

In Mobile Peer, Perst (2011) object-oriented library was used to store any IndexableDoc object (e.g. UserProfile, News Articles/Channels) for fast retrieval. It includes a basic full text index for fast searching. In addition, Perst library can run on any Java-enabled Phone.

5.3.3 **Query Service**

The Query Service is designed for advance searching function in JHPeer network. In PPNews, only Super Peer has implemented this component that is mainly used for searching news articles/channels, user profiles, user contexts between Super Peers. It currently supports Boolean Query.

In Mobile Peer, searching functions is mainly used for searching news articles provided by News Service and handled by Super Peer.

5.3.4 **Peer Monitor**

Peer Monitor is implemented to keep checking peer online status periodically. The Peer Monitor of Super Peer is responsible for monitoring and querying the status of Mobile Peer. The Peer Monitor of Mobile Peer only has to respond to the query from Super Peer.
5.4 Implementation of News Service

The News Service is a 3rd party service on JHPeer framework that implements com.jhpeer.service.Service interface. This enables registering to Service Manager to receive system events (i.e. start, pause, and stop application). The News Service in Super Peer is made up with several sub-modules (and threads): **News Documents** and **Protocol, RSS fetcher, News Ranking** and sub-modules for each criterion. The News Service in Mobile Peer is a client-side service which is used to submit user’s profile/queries and receive information from Super Peer.

5.4.1 News Documents

News Document is used in News Service. It includes **NewsUserSetting**, **NewsUserProfile**, **FeedChannel**, **FeedItem**, and **FeedScore**.

The **NewsUserSetting** is a user profile module that holds the user settings for News Service including News channels subscription, number of news per category, user AHP weights preference, passed time period of news, and delivery preference. The XML Schema is shown in Figure 5.2.

![Figure 5.2 XML Schema of NewsUserSetting](image)

**NewsUserProfile** holds system generated user profile on News application. The XML Schema is shown in Figure 5.3.
FeedChannel holds the basic information about the news source that includes title, description, main page link, RSS link, and so on. The XML Schema of FeedChannel is shown in Figure 5.4.
FeedItem (Figure 5.5) holds the data of a news story that includes Feed Channel ID, title, description (content), authors, and so on.

FeedScore (Figure 5.6) holds the user votes (ratings) of News Items. It also helps to record reading history of the user.
Figure 5.7 News Context Model

Figure 5.7 illustrates the relationship between news documents. The common objects Peer, Context, User and Device (i.e. ContextAdvertisement, UserProfileAdvertisement, and DeviceAdvertisement) are defined in JHPeer framework. The News document extends these common objects. The NewsUserSetting and NewsUserProfile is part of UserProfileAdvertisement that extends UserProfileModule which is described in section 3.3.1.3. FeedScore is a type of user activity which indicates user reading operations.

5.4.2 News Service Protocol

A protocol is developed to enable the communication between News Services on Mobile Peer and Super Peer. The news service protocol is designed to use a single message – FeedOperationMessage (XML schema is shown in Figure 5.8). Currently, the FeedOperationMessage has 6 types of operations: INIT, REGISTER, POST, GET, SEARCH, and PUSH.
**INIT** operation is implemented to initialise the news service in Mobile Peer that requests Super Peer to send the basic information to Mobile Peer such as new channel subscription list, and FeedChannel information.

**REGISTER** is used to submit user profile to Super Peer and register for receiving pushed message.

**POST** operation is used to submit FeedScore document when user votes for a news item or user read a news article.

**GET** Operation is used to retrieve FeedChannel information and load more news articles when user is willing to read more. The News service usually delivers top-k (i.e. 10) news based on user settings. However, the mobile news service can change the settings and request Super Peer to send more news (i.e. top-20) when user requests on user interface.

**SEARCH** operation handles user search query.

**PUSH** operation is used to push message from Super Peer.

The FeedOperationMessage contains an **MsgType** variable which is defined to identify the type of message. 14 different values are defined for different types of messages. For example, if the value of **MsgType** is 64, it indicates the message is a push message. If the value of **MsgType** is 8, it is POST request message. The **OperationResult** holds the response message including error messages, while the other variables such as **InputPipeAdv**, **NewsUserSetting**, **UserInfo**, **FeedChannel**, etc are used to hold the related documents and information that is used in different operations. For example, **INIT** operation requires Mobile Peer to submit **InputPipeAdvertisement**. This enables Super Peer to communicate to Mobile Peer based on the given **InputPipeAdvertisement** for returning news channel subscription list and FeedChannel documents.
5.4.3 RSS Aggregator

RSS Aggregator is a sub-module that runs in the background to feed news articles from various sources. The procedure of RSS aggregator is designed as follows:

1. Load all FeedChannel documents from Cache Service, and carry on for each channel.
2. Check timestamp of last feed, if it is greater than time interval, go next step.
3. Fetch news articles from RSS.
4. Check for redundant news articles, store or update news articles in Cache.
5. Generate FetcherEvent to notify arrival of new items.

The RSS Aggregator is implemented using ROME (ROME, 2010) library which is an open source library for parsing different version of RSS and Atom feeds.

5.5 News Ranking

The design of News Ranking is based on the AHP-MCR approach described in section 4.2. It follows the 4 steps: 1) Construction of criteria hierarchy, and the
alternatives, 2) Constructions of criteria comparison pairwise matrix, 3) Derivation of weight of criteria from comparison pairwise matrix, 4) Computation of priority for each alternative. The details of each step are described in the following sections.

5.5.1 Construction of Criteria Hierarchy, and the Alternative

The criteria selected for PPNews is shown in Figure 5.9, which is based on 4 identified criteria: Profile, Situation, Rates, and Attributes. The sub-criteria are selected based on the common ranking factors in News domain. It is also suitable for a new News recommender that does not require large amount of historical data.

![Figure 5.9 PPNews AHP Hierarchy](image)

5.5.1.1 Profile Criterion

Profile Criterion is the key factor for News Ranking due to frequent updating of news web site. To rank new articles, the long-term and short-term user histories should be analysed (H. J. Lee & Park, 2007; J. Liu, Dolan, & Pedersen, 2010; Papadogiorgaki et al., 2008). In PPNews, the long-term factor is the particular news category that user is interested in. For the short-term factors, news keywords are used to rank the news stories. Only keywords of recently read news are used, since it may change over time (J. Liu et al., 2010).
The score of profile criterion is computed by the combined score of all sub-criteria and their weights using equation (4.3). The sub-criteria include User Interest and Keywords.

5.5.1.2 User Interest Criterion

In PPNews, a Bayesian Network (BN) is developed to predict the level of interesting news categories for a particular user and handle the problem of cold-start for new user. CF/CBF techniques require analysing a large amount of users’ history. The BN is constructed via a summarisation of empirical data from recently published researches (Dutton, Helsper, & Gerber, 2009; Mintel, March 2009) in news domain. The BN is developed based on the methods described in section 3.5.1.

5.5.1.2.1 Construction of BN

The selection of relevant variables for PPNews is based on theoretical and empirical study in news domain. By reviewing of the major news web sites and published research works (Billsus & Pazzani, 2000, 2007; Chu et al., 2009; Claypool et al., 1999; Dutton et al., 2009; H. J. Lee & Park, 2007; Mintel, March 2009), five variables were identified as follows:

- Gender: The preferred news is varied because of user’s gender (Chu et al., 2009; Dutton et al., 2009; Mintel, March 2009). For instance, a man is more interested in sport than a woman. Obviously, two states are used for this variable: Male and Female.

- Age Range: It is a common demographic factor that affects the choice of news category. Different age groups have different preferred news category (Chu et al., 2009; Dutton et al., 2009; Mintel, March 2009). For this variable, common age ranges are used: 0-24, 25-34, 35-44, 45-54, 55-64, 65+.

- Occupation Status: Working status is an effect on identifying user interests (Dutton et al., 2009; Mintel, March 2009). For this variable, four basic occupation states are considered: Full-Time, Part-Time, Unemployed, and Retired.
- **Social Economic Grade**: The Social Economic Grade is influential in the category of news users preferred (Dutton et al., 2009; Mintel, March 2009). For this variable, four states are considered: AB (£35-50k+), C1 (£25-35k), C2 (£15-25k), DE (£0-15k).

- **Category (News)**: There are many ways to organise news articles. For simplicity, this research only considers the news at the top level rather than organising them into different levels of hierarchy. The category is organised in common states (Billsus & Pazzani, 2000; Claypool et al., 1999; H. J. Lee & Park, 2007; Mintel, March 2009): Headlines, Business, Politics, Entertainment, Health, Science, Technology, Sports, World, UK (nation).

The News BN is developed and the relationships between the variables are established as shown in Figure 5.10. The values in the CPT for each variable are assigned based on the summarisation of empirical data from (Dutton et al., 2009) and (Mintel, March 2009). Four variables: Gender, OccupationStatus, AgeRange and SocialEconomicGrade are root nodes, which influence the News Category. If the probability distribution of any of the three variables is changed, the probability distribution of the interested news category also changes.

![Figure 5.10 Prior Probability Distributions of News BN](image-url)
5.5.1.2.2 Predicting User Interest Category

The posterior probability distribution of the Category node is calculated using given values from users’ demographic contexts. A user’s news interests category is predicted based on the user’s personal information (Gender, Occupation Status, Age, and Social Economic Grade). Figure 5.11 shows the predicted result of posterior probability distribution for Category node given the user’s profile (Gender=Male, Age Range=0-24, Occupation=PartTime, Social Economic Grade=C1), which is computed based on the chain rule of probability theory defined in equation (3.1). On the other hand, if the user is not willing to provide any demographic contexts for privacy reasons, the prior probability distributions of News BN will be used.

![Digest of Figure 5.11 Posterior Probability Distributions](image)

When a user’s past reading history is accumulated, the soft evidence method is used to predict the user’s news interest. Figure 5.12 shows an example of applying the soft evidence.
Based on a user’s past click behaviour, the probability of Category (C) node with soft evidence (C’) is calculated as follows:

From Figure 5.11, \( P(C = \text{Headlines}) = 0.15 \), and given 21% of soft evidence for Headlines variable. By using Bayes’ Theorem to perform the inference, it is equal to:

\[
\text{Bel}(C' = \text{Headlines}) = \alpha P(C' = \text{Headlines}|C = \text{Headlines})P(C = \text{Headlines})
\]

\[
= \alpha \times 0.15
\]

Since \( \text{Bel}(C' = \text{Headlines}) + \text{Bel}(C' = \text{UK}) + \text{Bel}(C' = \text{World}) + \cdots + \text{Bel}(C' = \text{Health}) = 1 \), this gives us \( \alpha = 9.3213 \). Therefore, \( \text{Bel}(C' = \text{Headlines}) = 9.3213 \times 0.21 \times 0.15 = 0.2936 \), as shown in Figure 5.12.

By applying the soft evidence in the BN, personalised posterior probability distributions of Category node can be generated. This posterior probability distribution will be used to define the priority of User News Interest Criterion \( r_{ij} \).

In PPNews, the structure of BN is fixed and the data is complete. The News BN is revised based on parameter learning (Korb & Nicholson, 2004) in Super Peers.
5.5.1.3 Keywords Criterion

Keywords are a list of words that are used as a short-term factor for news ranking. The keywords list is generated based on users’ activities in past 7 days. In PPNews, Keyword-based searching techniques are designed to compute the score of news stories. Based on users’ past activities, all words in the clicked news title will be indexed. Only top-k (i.e. top-10 is used by default) keywords are used in the scoring process. The scoring process is computed using Term Frequency-Inverse Document Frequency (TF-IDF) which is shown as follows:

$$w_{m,n} = tf_{m,n} \cdot idf_m , \quad tf_{m,n} = \frac{n_{m,n}}{\sum_k n_{k,n}} , \quad idf_m = \log \frac{N}{n_m}$$  \hspace{1cm} (5.1)

where $n_{m,n}$ is the number of occurrences of the considered term, $t_m$, in news story $(n)$, and the denominator is the sum of number of occurrences of all terms in news stories. $idf$ takes the logarithm of the number of all documents, $N$, over the number of documents containing the term $n_m$. The priority of a news article $(r_{i,j})$ is equal to the sum ($\sum_m w_{m,n}$) of the keywords’ weights ($w_{m,n}$).

In addition, WordNet (Miller, 1995)-based query expansion is used to expand the original query in order to incorporate related keywords. For example, “house” can be expanded to “house, flat, living”. The weight of query keyword is multiplied by a boost factor (i.e. 0.9) which distinguishes from original keywords by decreasing the weight of expanded keywords.

Furthermore, global keywords are applied in order to provide optimised results to new users.

5.5.1.4 Situation Criterion

Situation criterion currently includes location criterion and weather criterion. By using Geocoding services in Server peer, it converts latitude and longitude which is obtained from a device’s GPS, to textual location (address) for carrying out term weighting process on the news articles. For example, if a user is located in Portsmouth, Hampshire, UK, the system will go through similar process as
Keywords Criterion by using TF-IDF. The terms used in this example will be Portsmouth, Hampshire, and UK. If a news article contains these terms, it will get a higher score. Moreover, other situation contexts may be relevant in News domain such as weather contexts. For instance, Weather RSS feed can be used to retrieve textual weather information (e.g. flooding) using user’s location. The TF-IDF value of each keyword in weather information is calculated. If a news story contains the weather keywords (e.g. flooding), it will be given a higher priority.

5.5.1.5 Rates Criterion

Rates Criterion is used to determine the importance of a news story. Currently, click-rate of news items is implemented. It is based on simple counting mechanism. Higher scores are applied to most clicked items.

5.5.1.6 Attributes Criterion

Attributes criterion refers to time recency in PPNews since only pre-defined RSS websites is currently allowed. The time recency of news article is computed as follows (adopted from (H. J. Lee & Park, 2007)):

\[ r_{i,j} = -\left( \frac{t_{\text{present}} - t_{\text{posted}}}{t_\beta} \right)^2 + 1 \]  \hspace{1cm} (5.2)

where \( t_{\text{present}} \) is the current timestamp, \( t_{\text{posted}} \) is when the article was posted and \( t_\beta \) is the base timestamp which is equal to 24 hours. For instance, if an article is posted at 13:00 GMT and current time is 15:00 GMT, the weight is equal to \(-((15-13)/24)^2+1 = 0.9931\). The score will be zero when the posted time is greater the base time (\( t_\beta \)). The decreasing rate of \( r_{i,j} \) is shown in Figure 5.13.
5.5.1.7 **The Alternatives**

The alternatives in this ranking process are the news articles which are fetched from RSS Aggregator or Caching Component.

5.5.2 **Construction of Criteria Comparison Pairwise Matrix**

The criteria comparison pairwise matrix is implemented based on individual-based approach that combines both explicit (user preference) and implicit (automatic learning through user activity) methods to enable automatic priority adjustment. Only the weight of the four major criteria has been made available for users to adjust on the GUI because a large number of questions may discourage some users. The importance of sub-criteria is assigned a default value (i.e. 1) which refers to equal importance.

The computation is operated from the bottom level to top level of the criteria hierarchy. The following table shows the computation of level-2 CPM in criteria hierarchy with assumption that bottom level has been computed.

Table 5.1 gives a sample data set of user history with the scores of criteria.
Table 5.1 Sample Dataset for CPM Computation

<table>
<thead>
<tr>
<th>Profile</th>
<th>Situation</th>
<th>Rates</th>
<th>Attributes</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.6</td>
<td>0</td>
<td>0.4</td>
<td>0.3</td>
</tr>
<tr>
<td>0.2</td>
<td>0.6</td>
<td>0.9</td>
<td>0.7</td>
</tr>
<tr>
<td>0.7</td>
<td>0.6</td>
<td>0.8</td>
<td>0.5</td>
</tr>
<tr>
<td>0.4</td>
<td>0.9</td>
<td>0.4</td>
<td>0.1</td>
</tr>
<tr>
<td>0.6</td>
<td>0.4</td>
<td>0.1</td>
<td>0.2</td>
</tr>
<tr>
<td>0.3</td>
<td>0.7</td>
<td>0.3</td>
<td>0.6</td>
</tr>
<tr>
<td>0.3</td>
<td>0.3</td>
<td>0.8</td>
<td>0.9</td>
</tr>
<tr>
<td>0.4</td>
<td>0.5</td>
<td>0.3</td>
<td>0.2</td>
</tr>
<tr>
<td>0.6</td>
<td>0.6</td>
<td>0.5</td>
<td>0.8</td>
</tr>
<tr>
<td>0.7</td>
<td>0.1</td>
<td>0.4</td>
<td>0.9</td>
</tr>
</tbody>
</table>

Table 5.2 shows the computation of numerical judgement of four major criteria that combines both user assignment ($U_i$) and automatic learning ($A_i$). The automatic learning is computed using the user history which is shown in Table 5.1. The judgement is computed using the equation (5.3) where $N$ is all criteria, $n$ is sub-set of $N$, and $v_{i,n}$ is the value of Criterion $i$ in the sub-set $n$. The judgement is equal to the total number of criterion $i$ that has the highest value among the sub-set $n$ criteria. For example, the sub-set $n$ is P, S, R, A, the value of P criteria has 2 highest value among four criteria in the dataset. Based on the equation (5.3), $A_P$ is equals to 2. And, $A_P + A_S + A_R + A_A$ is equals to total number of user history which is 10.
\[ A_i = \left( \frac{\text{count} \left( \max_{n \in N} v_{i,n} \right)}{\text{total}} \right)^{10} \]  
\[ (5.3) \]

The final numerical judgement of criterion \( C_i \) is the average value of \( U_i \) and \( A_i \).

**Table 5.2 Computation of Numerical Judgement of Criteria**

<table>
<thead>
<tr>
<th>Criterion Comparison</th>
<th>User assigned value ( (U_i) )</th>
<th>Automatic learning ( (A_i) )</th>
<th>Final value ( (C_i) )</th>
</tr>
</thead>
<tbody>
<tr>
<td>Profile(P-S-R-A)</td>
<td>4</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>Situation(P-S-R-A)</td>
<td>7</td>
<td>3</td>
<td>5</td>
</tr>
<tr>
<td>Rates(P-S-R-A)</td>
<td>3</td>
<td>2</td>
<td>2.5</td>
</tr>
<tr>
<td>Attributes(P-S-R-A)</td>
<td>6</td>
<td>3</td>
<td>4.5</td>
</tr>
<tr>
<td>Situation(S-R-A)</td>
<td>5</td>
<td>4</td>
<td>4.5</td>
</tr>
<tr>
<td>Rates(S-R-A)</td>
<td>1</td>
<td>3</td>
<td>2</td>
</tr>
<tr>
<td>Attributes(S-R-A)</td>
<td>3</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>Rates(R–A)</td>
<td>5</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>Attributes(R–A)</td>
<td>3</td>
<td>5</td>
<td>4</td>
</tr>
</tbody>
</table>

Table 5.3 shows the construction of CPM using the \( C_i \) from above table. The CPM for each criterion pair \( i \) and \( j \) is equal to \( p_{ij} = C_i / C_j \). For example, \( p_{PS} = C_P / C_S = 3 / 5 \). After \( p_{ij} \) is filled, the value will be normalised. The normalised \( p'_{pp} \) is computed as:

\[ p'_{pp} = \frac{p_{pp}}{p_{pp} + \frac{1}{5} + \frac{1}{3} + \frac{1}{5}} = \frac{1}{\frac{5}{3} + \frac{2.5}{3} + \frac{4.5}{5}} = \frac{1}{5} = 0.2. \]
Table 5.3 Construction of CPM

<table>
<thead>
<tr>
<th>CPM</th>
<th>P</th>
<th>S</th>
<th>R</th>
<th>A</th>
</tr>
</thead>
<tbody>
<tr>
<td>P</td>
<td>1</td>
<td>3/5</td>
<td>3/2.5</td>
<td>3/4.5</td>
</tr>
<tr>
<td>S</td>
<td>5/3</td>
<td>1</td>
<td>4.5/2</td>
<td>4.5/3</td>
</tr>
<tr>
<td>R</td>
<td>2.5/3</td>
<td>2/4.5</td>
<td>1</td>
<td>5/4</td>
</tr>
<tr>
<td>A</td>
<td>4.5/3</td>
<td>3/4.5</td>
<td>4/5</td>
<td>1</td>
</tr>
</tbody>
</table>

\[
\text{normalise}
\]

5.5.3 Derivation of Weight of Criteria from Comparison Pairwise Matrix

In this step, the weight of criteria \( w_i \) is calculated. By using the CPM from previous step, the weight of criteria \( w_i \) by geometric mean \( w_i = \left( \prod_{j=1}^{n} p_{ij} \right)^{1/n} \) is calculated as follows:

Profile \( (w_P) \): \( (0.2 * 0.22 * 0.23 * 0.15)^{1/4} = 0.20 \)

Situation \( (w_S) \): \( (0.33 * 0.37 * 0.43 * 0.34)^{1/4} = 0.37 \)

Rates \( (w_R) \): \( (0.17 * 0.16 * 0.19 * 0.28)^{1/4} = 0.20 \)

Attributes \( (w_A) \): \( (0.3 * 0.25 * 0.15 * 0.23)^{1/4} = 0.23 \)

5.5.4 Computation of Priority for Each Alternative

In order to compute the priority for each alternative, two procedures are followed. Firstly, normalisation of local scores is carried out to eliminate the differences between local scores. Secondly, final score is calculated from the bottom-level in criteria hierarchy. The following shows an example of ranking news articles:

Table 5.4 shows the given local scores of four criteria (i.e. Profile, Situation, Rates, and Attributes) for 5 news articles. The local scores will be normalised before compute the final score. The normalised local score \( N_{ij} \) is calculated by dividing the
local score by the maximum local score \((r_k)\) in criterion \((N_{i,j} = \frac{r_{ij}}{\max_k r_k})\). For instance, News A: \(N_{A,P} = \frac{1.2}{2.1} = 0.57\).

Table 5.4 Computation of Local Score of News Articles

<table>
<thead>
<tr>
<th>News Articles</th>
<th>P</th>
<th>S</th>
<th>R</th>
<th>A</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>1.2</td>
<td>0.2</td>
<td>10</td>
<td>0.6</td>
</tr>
<tr>
<td>B</td>
<td>2.1</td>
<td>0.3</td>
<td>5</td>
<td>0.4</td>
</tr>
<tr>
<td>C</td>
<td>0.9</td>
<td>0.8</td>
<td>13</td>
<td>0.5</td>
</tr>
<tr>
<td>D</td>
<td>1.7</td>
<td>0.5</td>
<td>15</td>
<td>0.7</td>
</tr>
<tr>
<td>E</td>
<td>0.8</td>
<td>0.01</td>
<td>30</td>
<td>0.3</td>
</tr>
</tbody>
</table>

normalise

The final score \((score_i)\) of each alternative is equal to the sum of normalised local scores \((N_{i,j})\) multiplied by the weight of criterion \((w_j)\) that is written as 
\[
score_i = \sum_{j=1}^{n} N_{i,j} w_j
\]
and is shown in the following table:

Table 5.5 Computation of Final Score

<table>
<thead>
<tr>
<th>News Articles</th>
<th>Final Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>0.47 (0.57 * 0.20 + 0.25 * 0.37 + 0.33 * 0.20 + 0.86 * 0.23 )</td>
</tr>
<tr>
<td>B</td>
<td>0.51</td>
</tr>
<tr>
<td>C</td>
<td>0.71</td>
</tr>
<tr>
<td>D</td>
<td>0.73</td>
</tr>
<tr>
<td>E</td>
<td>0.38</td>
</tr>
</tbody>
</table>
Therefore, the order of News articles is D, C, B, A, E.

5.6 User Interface

PPNews is designed to run on any Java-based mobile platform. An adaptable graphical user interface (GUI) is implemented that can be run on a variety of mobile devices such as Java-enabled mobile devices, Android devices, and BlackBerry phones (as shown in Figure 5.14). PPNews GUI uses the Lightweight User Interface Toolkit (LWUIT)\(^2\). LWUIT supports many modern features (e.g. layout, rich widgets themes, localisation, touch screen) and have adapted to several mobile platforms (e.g. Android, BlackBerry).

![Figure 5.14 PPNews Screenshots on BlackBerry, Android and Standard Java Emulators](https://lwuit.dev.java.net/)

The recommendations are pushed to a mobile device via JHPeer network. Figure 5.15 shows the main screen of PPNews that includes clickable buttons in a grid layout. The button *News* is the news recommendation service. The button *Status* is to show the current network status and also show information from context provider such as current user location. The *Setting* interface allows a user to set

\(^2\) [https://lwuit.dev.java.net/](https://lwuit.dev.java.net/)
user preference (i.e. demographic, email), general applications (i.e. push period, enable vibration, enable HTML, enable GPS), and news service settings (i.e. news subscription, weight of AHP criteria).

![Image](image.png)

**Figure 5.15 Main UI on Android Phone**

When a user uses the News service for the first time, a wizard UI (Figure 5.16) will be displayed to ask user to provide general preference, basic information, AHP weights, and News channel subscription list.

After the user finishes the settings, News items will be pushed from Super Peer to Mobile Peer automatically. News category UI (Figure 5.17) will be shown and the number of unread news for each category will be located beside it. By clicking one of the News categories, a list of News items which are ordered by the computed scores will be displayed. Unread items will be highlighted in BOLD. Users can re-sort the list of items by clicking Sort menu from left corner to select the available criteria. Sorting is based on scores of 4 major criteria which have been embedded in the metadata of pushed items. Finally, content of News will be displayed by clicking on the list. Users can provide ratings to the News or read the full articles from original web pages by clicking the menu buttons at the left and right corners.
Figure 5.16 News Wizard UIs

Figure 5.17 News Category (left), List of News (middle), and News Display (right)
5.7 Summary

PPNews is built on top of JHPeer framework that incorporates user contexts and news article attributes for proactive personalised news recommendation. All JHPeer components are implemented in a modular way which can be extended and reused by other mobile applications. The major benefit of using JHPeer is to allow mobile devices listening for “push” News Items. The delivery of news item is faster than using traditional push-pull methods. The BN-based user profiling method is applied to estimate a user’s preference and construct user profile for new users, which effectively handles the cold-start problem. The AHP-MCR approach is used to rank news articles based on a user’s preference, past click history and news attributes. The temporal attributes, e.g. location, news recency, are up-dated in real time. Users can customise the system settings and easily interact with the system through a GUI. Based on the ranking and user settings, PPNews is capable of providing personalised recommendation of new news articles for both existing and new users. The implementation of PPNews comprises a set of software components that can be reused in other applications. The developed system can run on any Java-enabled platform such as Android and Blackberry.
Chapter 6
Experimental Evaluation

To evaluate the feasibility and technical deployment of JHPeer framework and the effectiveness of the AHP-MCR approach, a real application PPNews recommendation system is implemented. This chapter describes the experimental setup and reports the empirical results.

6.1 Experimental Setup

The analysis of the JHPeer framework mainly focuses on mobile peer, since the user experience is more important than Super Peer. Super Peers are mainly deployed on Desktop/Server computers which have extensive resources including powerful processing and most likely persistent broadband connectivity. The performance of a Super Peer is related to the design of services. Besides, some research works have already been carried out to examine the scalability and performance of JXTA on Desktop and Mobile environment reported in (Antoniou, Cudennec, Jan, & Duigou, 2007) and (Satish Narayana, 2008) respectively. This experiment primarily analysed the performance on mobile information delivery to show the usefulness of a proactive recommendation application.

The evaluation was carried out by using two different handsets: HTC Desire Smartphone (Android 2.2, 1GHz CPU, 576MB RAM, HSPA/WCDMA/GSM, Wi-Fi 802.11b/g, 3.7-inch SLCD Screen, Brightness LOW, Network Location Service), and Google Galaxy Nexus (Android 4.0.2, 1.2 GHz Dual-Core CPU, 1 GB RAM, HSPA/GSM, Wi-Fi 802.11b/g/n, 4.65-inch Super AMOLED, Brightness LOW, Network Location Service).

The application size of PPNews is 1624KB (without obfuscation). The memory usage of PPNews is about 28MB, as shown in Figure 6.1. The Android Platform was selected since it can retrieve detailed battery status and it is a relatively powerful mobile computing platform. On the other hand, the Super Peer is installed on a
Laptop with Intel Core 2 T7200 CPU (2.00 GHz, 4 MB Cache, 667 MHz FSB) and 3 GB of memory that runs on Windows 7 SP1 (64-bit) and Java 6.

![Image](image.png)

**Figure 6.1 PPNews Memory Usage**

In this experiment, PPNews was configured to push random news article to the mobile device continuously in order to evaluate the performance of JHPeer platform in different settings. The settings include: Wi-Fi with Screen OFF, Wi-Fi with Screen ON, Cellular Network with Screen ON, and Cellular Network with Screen OFF. The Cellular Network used was O2 UK and was tested indoors, and the broadband of Super Peer is provided by British Telecom. It was necessary to test the performance for both Screen ON and Screen OFF, since the new generation of smartphones will manage and lower the power consumption when screen is turned off. The CPU frequency and voltage can change dynamically depending on the system status. The second reason is because JHPeer framework will also pause or stop unessential modules/services (i.e. Context Providers) to preserve battery.

### 6.1.1 Experiment Result

The experiment was run for 3 days for each handset in order to complete all 4 environment configurations. Each test started with 100% battery and stopped when the phone powered off automatically. One random news article was pushed from
Super Peer to Mobile Peer during each operation. A report message was sent back to Super Peer including the log information such as time used on message handling. Several types of data were collected by Super Peer during the test as follows.

6.1.1.1 Messages Transmission Result

Figure 6.3 shows the accumulated number of received news articles before the battery ran out. The results show that performance on Wi-Fi network with Screen ON is the best in HTC Desire with 336758 articles. However, Wi-Fi network with Screen OFF got the best result in Google Galaxy Nexus with 365989 articles. The reason for this is because Android system in HTC Desire reduces the CPU frequency when the screen is switched off to save battery. The slowed processing speed leads to a longer message handling time (refer to Table 6.2). It is 7 times slower to complete one request-response operation (refer to Table 6.2). The system running time is just 4 times longer than Wi-Fi Screen ON (refer to Figure 6.4). Therefore, the battery is wasted in Wi-Fi idle time.

In Google Galaxy Nexus, the result is similar to HTC Desire before 66% of remaining battery. However, the Android system changed its behaviour and increased the CPU frequency (refer to Figure 6.5 that data rate increased after 11 hours 30 minutes). With high processing power and Screen OFF (i.e. longer battery life), the result is better than Wi-Fi Screen ON.

For Network Screen ON/OFF, in HTC Desire, the result is similar to Wi-Fi environment that Screen OFF reduces the performance. In Google Galaxy Nexus, the overall performance is much better than HTC Desire, since it is embedded latest chipsets. In the experimental, the Google Galaxy Nexus is running at high CPU frequency in both Screen ON and OFF (refer to Figure 6.5 high data transmission rate in Screen OFF). In this situation, the battery consumption of large screen affects the battery life. Therefore, less messages is able to transmit in Network Screen ON.

In conclusion, the newer smartphone performs better than the old one. The modern operating system intelligently manage the system (i.e. balance the power
consumption for different hardware components) to provide best performance in devices’ battery lifespan. In this experiment, it shows that JHPeer framework is capable of transmitting large amounts of messages between peers during the limited battery lifespan of mobile devices. The performance of using JHPeer framework to deliver message is increasing with advanced mobile devices. In addition, the performance of JHPeer is acceptable to be used in an old smartphone.
6.1.1.2 Data Transmission Result

Figure 6.4 and Figure 6.5 shows the details of data transmitted for each 10 minutes. Besides, the total running time of battery lifespan is shown in the figures. The average data transmission in each 10 minutes for HTC Desire in Wi-Fi-S-OFF was 3961KB, Wi-Fi-S-ON was 30125KB, Network-S-OFF was 2519KB, Network-S-ON was 2492KB; for Google Galaxy Nexus Wi-Fi-S-OFF 5486KB, Wi-Fi-S-ON was 14434KB, Network-S-OFF was 2030KB, Network-S-ON was 2009KB.

From the figures, it shows that the average data transmission rate of HTC Desire is higher than Google Galaxy Nexus. However, the total running time of HTC Desire is less than Google Galaxy Nexus. This is because the Google Galaxy Nexus is embedded with a dual-core CPU, but the application of this test is written in single thread which means it has to complete a request-response operation before it starts another operation. Since only one core is busy, the Android system will manage the power consumption that another core will run in low frequency to save battery. In addition, the new CPU cost less power than the old CPU in HTC Desire. Therefore, the overall running time of Google Galaxy Nexus is longer than HTC Desire.

In addition, Table 6.1 shows the total data transmitted during the tests. The result shows that Google Galaxy Nexus can transmit more data in screen on environments, while HTC Desire can transmit more data in screen off environments. The main reason is because the screen of Google Galaxy Nexus consumes more power battery and the intelligence power management in Android system that CPU frequency can change dynamically (depend on system load) in both Screen On and Off. On the other hand, the power management in HTC Desire is set to run in low frequency when screen is off.

In conclusion, the Wi-Fi network can transmit more data than 3G cellular network in the battery lifespan and the power consumption is low where the battery life can last for 5 hours, even when it is receiving messages continuously. While in the normal case, the battery could last two days or more.
Figure 6.4 Received KB in Each 10 minutes (HTC Desire)

Figure 6.5 Received KB in Each 10 minutes (Google Galaxy Nexus)

Table 6.1 Total Data Transmitted during Battery Lifespan

<table>
<thead>
<tr>
<th>Mode</th>
<th>HTC Desire</th>
<th>Google Galaxy Nexus</th>
</tr>
</thead>
<tbody>
<tr>
<td>Wi-Fi-S-OFF</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Wi-Fi-S-ON</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Network-S-OFF</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Network-S-ON</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

100
### 6.1.1.3 Average Time Consumption per Operation

The following tables show the average time (in millisecond) to deliver news to the mobile device. The average message handling time represents the time used to process the news article from XML string to news item object. The average sending time of a response message represents the time used to send a reply message to a server for notifying receipt and logging record.

#### Table 6.2 Average Time Used in System Operation (HTC Desire)

<table>
<thead>
<tr>
<th></th>
<th>Average Message handling time</th>
<th>Average sending time of response message</th>
<th>Average operation completed time</th>
</tr>
</thead>
<tbody>
<tr>
<td>Wi-Fi-S-ON</td>
<td>9.89ms</td>
<td>2.58ms</td>
<td>50.71ms</td>
</tr>
<tr>
<td>Wi-Fi-S-OFF</td>
<td>29.35ms</td>
<td>3.5ms</td>
<td>370.23ms</td>
</tr>
<tr>
<td>Network-S-ON</td>
<td>22.48ms</td>
<td>4.31ms</td>
<td>642.25ms</td>
</tr>
<tr>
<td>Network-S-OFF</td>
<td>22.03ms</td>
<td>4.12ms</td>
<td>641.32ms</td>
</tr>
</tbody>
</table>

#### Table 6.3 Average Time Used in System Operation (Google Galaxy Nexus)

<table>
<thead>
<tr>
<th></th>
<th>Average Message handling time</th>
<th>Average sending time of response message</th>
<th>Average operation completed time</th>
</tr>
</thead>
<tbody>
<tr>
<td>Wi-Fi-S-ON</td>
<td>9.89ms</td>
<td>2.58ms</td>
<td>50.71ms</td>
</tr>
<tr>
<td>Wi-Fi-S-OFF</td>
<td>29.35ms</td>
<td>3.5ms</td>
<td>370.23ms</td>
</tr>
<tr>
<td>Network-S-ON</td>
<td>22.48ms</td>
<td>4.31ms</td>
<td>642.25ms</td>
</tr>
<tr>
<td>Network-S-OFF</td>
<td>22.03ms</td>
<td>4.12ms</td>
<td>641.32ms</td>
</tr>
<tr>
<td></td>
<td>Wi-Fi-S-ON</td>
<td>Wi-Fi-S-OFF</td>
<td>Network-S-ON</td>
</tr>
<tr>
<td>----------------</td>
<td>------------</td>
<td>-------------</td>
<td>--------------</td>
</tr>
<tr>
<td>Time (ms)</td>
<td>12.82ms</td>
<td>19.98ms</td>
<td>14.82ms</td>
</tr>
<tr>
<td>Latency (ms)</td>
<td>1.21ms</td>
<td>1.25ms</td>
<td>1.19ms</td>
</tr>
<tr>
<td>Delivery Time (ms)</td>
<td>64.17ms</td>
<td>169.08ms</td>
<td>454.37ms</td>
</tr>
</tbody>
</table>

From the results, it shows that Google Galaxy Nexus performs better than HTC Desire. The average time to deliver a news item (i.e. request-response operation) to mobile device was approximately 640 milliseconds for HTC Desire and 450 milliseconds for Google Galaxy Nexus in 3G cellular network, which are acceptable for a recommendation system.

### 6.1.2 Conclusion/Discussion

Throughout this experiment, it shows that JHPeer framework is suitable for developing context-aware information applications due to the following reasons:

- Capable of transmitting large number of messages which is useful to push Mobile Peer’s contextual information to Super Peer
- Reasonable time used on request-response operation
- Power consumption is low where the battery life can last for 5 hours, even when it is receiving messages continuously. While in the normal case, the battery could last two days or more.

In fact, only one news article is transmitted in each operation during the experiment. This contains lots of overhead to transmit and handle the messages that also includes unnecessary system logs. Therefore, it should have better performance in real usage (subjects to implementation and application domains). However, the current JHPeer framework is not suitable for developing time-sensitive applications such as Voice over IP (VoIP). Further investigation is required to implement a simplified messaging API in JXTA network layer. Alternatively, JXTA can be replaced by other P2P library.
6.2 Evaluation of AHP-MCR Approach

The user experience and effectiveness of the AHP-MCR approach is evaluated in terms of the accuracy of news rating. AHP-MCR approach is compared with the traditional recommendation techniques.

As no public dataset on context-aware recommendation is available, a dataset is created for the comparative study by inviting users. Twenty three users were participated in this experiment. RSS News web sites were built to allow user subscription. A feedback mechanism in PPNews application has been implemented. When the users run the PPNews application on their mobile phones (Android and Blackberry phone are used), news articles are automatically pushed to the users depending on their individual contexts and settings. User’s activities such as news clicking is automatically collected by Context Provider component and processed on Super Peer. Participants give ratings for news story on a 1 – 10 scale, which includes overall, profile, situation, rates, and attributes. 24 RSS News websites were defined in the application to allow user subscription. A result of 352 feedbacks was collected from participants.

6.2.1 Evaluation of Ranking Performance

There are numerous evaluation metrics for measuring recommender performance (Herlocker, Konstan, Terveen, & Riedl, 2004) such as Mean absolute error (MAE), Precision@Top-N, F-measure, ROC. Two popular metrics MAE and Precision@Top-N are used.

MAE

Mean absolute error method is used to measure the prediction performance by analysing the absolute error between the predicted score and the score which is given by user. The MAE is computed by summing all absolute errors between the predicted ratings and user ratings. Then it is divided by the total number of ratings. The lower the value, the better the result is. Equation (6.1) shows how MAE is calculated.
Standard user-based CF approach which uses cosine-based k-nearest neighbour method, and simple CBF approach which uses TF-IDF method were employed as a baseline. AHP-MCR approach is compared with other recommendation approaches (Adomavicius et al., 2005).

Table 6.4 summarises the MAE results of several recommendation approaches on PPNews dataset. The result shows that AHP-MCR approach performs slightly better than standard user-based CF and CBF techniques since it combines both CF and CBF techniques.

In addition, the predicted ratings of 4 major criteria are compared with user feedback ratings. The MAE of Profile, Situation, and Rates have similar performance. The MAE of Attributes is relatively good because time recency is easy for users to rate.

Table 6.4 Experimental Results of MAE

<table>
<thead>
<tr>
<th>Methods</th>
<th>MAE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Standard User-based kNN CF @ all users</td>
<td>0.9675</td>
</tr>
<tr>
<td>Standard User-based kNN CF @ 3 neighbourhoods</td>
<td>1.1261</td>
</tr>
<tr>
<td>Simple TF-IDF based CBF</td>
<td>0.9054</td>
</tr>
<tr>
<td>AHP-MCR</td>
<td>0.8504</td>
</tr>
<tr>
<td>Profile – Profile’ rating</td>
<td>0.8942</td>
</tr>
<tr>
<td>Situation – Situation’ rating</td>
<td>0.9159</td>
</tr>
<tr>
<td>Rates – Rates’ rating</td>
<td>0.9045</td>
</tr>
<tr>
<td>Attributes – Attributes’ rating</td>
<td>0.6908</td>
</tr>
</tbody>
</table>
Precision@Top-N

Precision@Top-N is used to measures the accuracy of the ranking algorithms in predicting Top-N items which user would rate highly. The Precision@Top-N is computed using the number of highly rated recommendations over the Top-N number of recommendations as shown in equation (6.2). The highly rated recommendations in this study are taken as any item which is rated with a score of 5 or more by the user.

\[
Precision@Top - N = \frac{\text{Number of highly rated recommendations}}{\text{Total number of recommendations}}
\]

(6.2)

Table 6.5 summaries the results of Precision@Top-5 and Precision@Top-10. In this experiment, the precision of each major criterion is calculated to show the effect of contextual criteria on the AHP-MCR ranking process. The precision is obtained using the average precisions of evaluated users. The AHP-MCR approach has better precision than standard user-based CF and TF-IDF based CBF. The standalone criterion performs relatively poorly especially for Situation and Rates criteria. This is because not all news articles contain the situation information and not all news articles in which a user is interested are highly clicked by other users.

<table>
<thead>
<tr>
<th>Methods</th>
<th>Precision@Top-5</th>
<th>Precision@Top-10</th>
</tr>
</thead>
<tbody>
<tr>
<td>Standard User-based CF @ all users</td>
<td>0.6511</td>
<td>0.6186</td>
</tr>
<tr>
<td>Standard User-based CF @ 3 neighbourhood</td>
<td>0.5845</td>
<td>0.5508</td>
</tr>
<tr>
<td>Simple TF-IDF based CBF</td>
<td>0.6042</td>
<td>0.5151</td>
</tr>
<tr>
<td>AHP-MCR</td>
<td>0.6993</td>
<td>0.6287</td>
</tr>
<tr>
<td>Profile</td>
<td>0.5915</td>
<td>0.5315</td>
</tr>
<tr>
<td>Situation</td>
<td>0.2942</td>
<td>0.2348</td>
</tr>
</tbody>
</table>
In conclusion, the study shows that using the AHP-MCR approach to embed context-aware computing can increase the quality of recommendation. However, there is only small improvement in terms of recommendation accuracy. The algorithms in recommendation systems are highly domain-specific. Other sophisticated algorithms include SVD/PLSA/linear regression-based CF algorithms and/or advanced CBF algorithms using natural language processing which can be considered in building AHP criteria when enough training data is acquired from users or from existing dataset.

6.3 User Experience

A survey was carried out to get statistical information on user satisfaction. A questionnaire was designed to measures four aspects of user satisfaction.

All participants who completed the experiment were invited to complete the survey.

Figure 6.6 presents the results of system effectiveness and accuracy from the first three questions. Most of the users provided positive feedback and only few users reported false positives. A user reported that number of uninterested news articles appeared at the top of the list. One of the reasons for this is that some news sources update short stories hourly to reflect the latest events. These news articles seem relevant to users. Currently, PPNews doesn’t have a criterion or filter for pushing only long stories (i.e. headlines) to users. PPNews will rank all stories from user subscriptions. Another problem that was detected was that some wrong keywords (noise keywords) were selected in the ranking process. These noise keywords were generated from the popular news articles. A quick and possible solution is to use natural languages processing techniques in the criteria such as using language-model based search engine – Indri (Trevor, Donald, Howard, & Croft, 2005).
Figure 6.6 Effectiveness and Accuracy

Figure 6.7 shows the result of user satisfaction on efficiency. Most of the users were satisfied with the response time of the system. But some were not as their phones performed slowly such as HTC G1, HTC Magic, etc. On the other hand, proactive pushing can reduce number of keystrokes. Unlike the RSS Readers, it does not require users to press and click any buttons for news deliveries. A few users felt that it did not reduce keystrokes as the user still had to navigate to news categories and access news items with the current UI designs.
Figure 6.8 shows the results of usability and layout of PPNews. PPNews application is easy to install on mobile phones by clicking the link. The users were assisted with the installation process. For the proactive pushing features, some users were not satisfied because of the frequent alerts. However, it can be disabled. Most users were happy with the UIs. As the UIs were developed to work on most Java-enabled mobile platforms, it is aware that the performance is not as good as using native UI libraries.

![Part 3: Usability and layout](image)

Figure 6.8 Usability and Layout

Figure 6.9 shows the user satisfaction with respect to connectivity, stability and reliability. Generally, the system was stable and reliable. PPNews application crashed at times due to a few programming bugs. And Mobile Peer was unable to connect to the Super Peer when users changed network interface such as Cellular Network to Wi-Fi. However, all these issues can be addressed through software updates.
The Part 5 of the questionnaire is user comments. Some of the comments have been discussed above. Other comments were related to enriching the features of PPNews applications such as save News for later, share with friends, and enable customised channels.

### 6.4 Discussion

Our study shows that using the AHP-MCR approach to embed context-aware computing can increase the quality of recommendations. The AHP-MCR approach performs better than single-criteria approaches. However, in the experiment, the participants have been shown how to use the system at the beginning. If users do not know how the criteria weighting works, it may affect the MAE results. Therefore, designing a questions-based interface with scale bar may help any new users. Moreover, the performance of AHP-MCR approach depends on the algorithms that are used in the criteria. The algorithms can be replaced at any time. More advanced algorithms such as PLSA can be applied when there have enough training data with user contextual information.
On the other hand, the user experience survey shows that users were generally satisfied with the PPNews performance although there is plenty of room to improve the application design such as UIs. Furthermore, PPNews may have the potential to be commercialised since hardly any proactive mobile recommenders with multi-criteria features exist in the current market.

6.5 Summary

This chapter has experimentally evaluated the JHPeer framework and AHP-MCR approach with a real news recommendation system. The experiments were divided into three parts: performance of the JHPeer network, evaluation of AHP-MCR method and user experience. The results show that the performance of message delivering and battery consumption is acceptable for developing information services on mobile devices. Further work on JHPeer would be optimising the performance, extending the functionality and supporting other platforms. For AHP-MCR method, the results show the method is effective although the improvement in recommendation accuracy is not significant. Applying more complex algorithms in the AHP-MCR ranking process and using large datasets to train the system would improve the performance. In addition, the system can use a suit of algorithms that are individually optimised for each domain to achieve high recommendation accuracy. Moreover, user experience survey also brought the opportunity of getting feedback from users about the system functionalities and performance. Users had raised a number of issues and enhancements that would be the future work for PPNews application.
Chapter 7
Conclusions and Future Work

7.1 Conclusions

With the readily available and affordable wireless and sensor technologies, proactive context-aware recommender is a potential solution to overcome the information overload and the common limitations of mobile device. Automatic provision of personalised information or recommendation tailored to each user’s needs/preferences will not only facilitate access to information but also remove barriers to the adoption of current and future services on mobile devices.

This dissertation focuses on developing a generic context-aware framework for supporting proactive personalised information recommendation in mobile environments. It has achieved the following results:

- It proposes a JXTA-based Hybrid Peer-to-peer framework for efficient organisation, representation, retrieval and management of context data, which enables rapid development of context-aware applications for mobile users. The JHPeer framework is based on hybrid P2P technology which enables scalability such that system can be grown by adding new servers. Mobile peers can communicate with each other directly. Unlike traditional push-pull mechanism, the JHPeer framework supports proactive information push. JHPeer is customisable and supports diverse higher level applications with a set of abstractions that are open to many possible implementations. Developers can incorporate their own mechanisms/algorithms and reuse/extend JHPeer APIs when implementing JHPeer components. Third party services and components can be added to JHPeer framework by using the given APIs. A context model is developed to model and represent mobile sensor data and contexts in order to share, distribute, store, index, transmit, and process contextual information in peer network. The above benefits
enable JHPeer framework to serve as a basis for future research or rapid application development.

- It develops an analytic hierarchy process based multi-criteria ranking approach to rate information and help users finding relevant items in various domains. AHP-MCR takes account of user context information. A general and extendable criteria hierarchy model is developed. The weights of the contexts criteria can be assigned by user or automatically adjusted via individual-based and/or group-based assignment.
- It develops a Bayesian Network (BN) based user profiling method to model users’ preference. The BN model construction process is defined. It is capable of handling the cold-start issue and can be applied in multiple applications.
- It designs and implements a proactive personalised news recommender on top of the JHPeer framework. PPNews is a concrete case study of the JHPeer framework, where all the JHPeer components are implemented and cooperated for news recommendation. The BN-based user profiling method is applied to estimate users’ preference. The AHP-MCR approach effectively rank news articles based on a user’s preference, past click history and news attributes. Unlike the traditional recommendation systems that usually require user to submit query explicitly, the PPNews automatically delivers news articles to mobile user based on the user’s interest and news content.

This thesis experimentally evaluates the JHPeer framework and AHP-MCR approach with the real news recommendation system. The results show that the performance of message delivering and battery consumption is acceptable for developing information services on mobile devices in JHPeer network. For AHP-MCR method, the results show the method is effective although the improvement in recommendation accuracy is not significant. User experience survey shows that users are generally satisfied with the PPNews performance although there is plenty of room to improve the application.
This thesis has demonstrated the effectiveness of the JHPeer framework with a real news recommender system and it can be applied to other application domains such as movie. This work has shown promising results in this direction and will help encourage more interests towards commercial implementation of personalised applications.

7.2 Future Work

Some of the directions in extending the JHPeer framework and its applications are discussed as follows.

- JHPeer framework can be enriched to support ontology technologies such as supporting RDF/OWL messaging, caching ontology-based information. One of the direction is to carry out research on fuzzy ontology (Bobillo & Straccia, 2008; Gómez-Romero, Bobillo, & Delgado, 2011) for handling and reasoning situational contexts in low computational mobile environments.

- Further Efforts would be spent on investigating the security and privacy mechanism in JHPeer to protect from unauthorised access to user contextual information and the system. This also enables users to control the sharing and the use of their contextual information that is acquired by the hidden sensors/reasoners.

- The proposed JHPeer framework is currently implemented using Java programming language. Although, Java is an OS independent language that can be executed in most of operating systems, it is not supported in newest mobile platforms such as Apple iOS, BlackBerry QNX. Therefore, it is essential to port JHPeer framework to any other programming languages in order to deal with the heterogeneous mobile environment. The network layer can be implemented by other lightweight frameworks in order to support time-sensitive application.

- Future effort would be spent on comprehensive evaluation of AHP-MCR approach in different application domains with other sophisticated algorithms such as singular value decomposition, probabilistic latent semantic analysis, linear regression based CF algorithms, etc.
References


116


121


ROME. (2010). ROME: RSS/Atom syndication and publishing tools, 2010, from https://rome.dev.java.net/


Appendix A  List of JHPeer Context Provider

<table>
<thead>
<tr>
<th>Context Provider Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>CellIDProvider</strong></td>
<td>The Cell ID Provider obtains the current mobile network ID including GSM, and CDMA.</td>
</tr>
<tr>
<td><strong>CellIDLocationProvider</strong></td>
<td>Use obtained Cell ID to detect approximate devices’ location.</td>
</tr>
<tr>
<td><strong>LocationProvider</strong></td>
<td>The Location Provider uses the platform location API to obtain user location.</td>
</tr>
<tr>
<td><strong>NetworkLocationProvider</strong></td>
<td>The Network Location Provider obtains location from platform network location service.</td>
</tr>
<tr>
<td><strong>PassiveLocationProvider</strong></td>
<td>The Passive Location provider listens to location from platforms’ location service. The location information may be obtained from any location service (i.e. GPS, Wi-Fi, IP location, Network) at any time.</td>
</tr>
<tr>
<td><strong>CalendarProvider</strong></td>
<td>The Calendar Provider fetches user events from platform calendar.</td>
</tr>
<tr>
<td><strong>WeatherProvider</strong></td>
<td>Based on user location, the Weather Provider fetches online weather information.</td>
</tr>
<tr>
<td><strong>PeerContextProvider</strong></td>
<td>The Peer Context Provider receives published contextual document from other peers.</td>
</tr>
<tr>
<td><strong>MobileSensorProvider</strong></td>
<td>The Mobile Sensor Provider accesses various sensors in the mobile device such as accelerometer, gravity, gyroscope, light, magnetic, pressure, proximity, rotation, and temperature sensors to detect corresponding sensor data.</td>
</tr>
<tr>
<td>Provider</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------</td>
<td>------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td><strong>BatteryProvider</strong></td>
<td>Obtain devices’ battery status.</td>
</tr>
<tr>
<td><strong>DeviceInfoProvider</strong></td>
<td>The Device Information Provider obtains platform and hardware information for generating DeviceProfileAdvertisement.</td>
</tr>
<tr>
<td><strong>PeerStatusProvider</strong></td>
<td>The System Status Provider provides information such as phone status (i.e. memory, network) and application status (i.e. uptime, running services)</td>
</tr>
<tr>
<td><strong>BluetoothProvider</strong></td>
<td>The Bluetooth Provider uses Bluetooth hardware to detect active neighbouring devices.</td>
</tr>
</tbody>
</table>
Appendix B   JHPeer Component Class Diagrams

Figure B.1 Context Service Class Diagram

Figure B.2 Context Provider Class Diagram

Figure B.3 Context Filter Class Diagram

Figure B.4 Context Listener and Context Discovery Listener Class Diagram
Figure B.5 IndexableDoc Class Diagram

```java
<<abstract>>
IndexableDoc

id:ID
dev_lastmod#y:long
dev_lifetime:long
dev_expiration:long

getID():ID
getIndexFields():String[]
getLastmodify():long
getLifetime():long
getExpiration():long
setLastmodify(long)
setLifetime(long)
setExpiration(long)
```

Figure B.6 Cache Class Diagram

```java
<<interface>>
Cache

getIndexableDocs(String):IndexableDoc[]
getIndexableDocs(String, int):IndexableDoc[]
saveIndexableDoc(IndexableDoc)
saveIndexableDoc(IndexableDoc, boolean)
updateIndexableDoc(IndexableDoc)
removeIndexableDoc(IndexableDoc)
removeIndexableDoc(IndexableDoc, boolean)
commit()

searchIndexableDocs(String, String, String):IndexableDoc[]
searchIndexableDocs(String, String, String, int):IndexableDoc[]
searchIndexableDocs(String, String[]):IndexableDoc[]
searchIndexableDocs(String, String[], String[]):IndexableDoc[]
searchIndexableDocs(String, String, String, int, CacheQueryResult, Filter, Sort)
searchIndexableDocs(String, String[], String[], int, CacheQueryResult, Filter, Sort)
```

Figure B.7 Query Service Class Diagram

```java
<<interface>>
QueryService

MODULE_CLASS_ID:ModuleClassID
MODULE_CLASS_NAME:moduleName
MODULE_CLASS_DESC:moduleDesc
DEFAULT_LIFETIME:long
DEFAULT_EXPIRATION:long

addQueryListener(QueryListener)
removeQueryListener(QueryListener):boolean
publish(IndexableDoc)
remotePublish(PeerID, IndexableDoc)
getLocalIndexableDocs(String, String, String):CacheQueryResult
getRemoteIndexableDocs(PeerID, String, String, String, int):int
getRemoteIndexableDocs(PeerID, String, String, String, int, QueryListener):int
getRemoteIndexableDocs(PeerID, String, String[], String[], int, QueryListener):int
removeIndexableDoc(IndexableDoc)
```

129
Figure B.8 Peer Monitor Class Diagram
Appendix C  JHPeer XML Scheme

Figure C.1 XML Schema of ContextAdvertisement

Figure C.2 XML Schema of Context_Location
Figure C.3 XML Schema of Context_CellId

Figure C.4 XML Schema of Context_Battery
Figure C.5 XML Schema of Context_Address
Figure C.6 XML Schema of DeviceProfileAdvertisement
Figure C.7 XML Scheme of UserProfile

Figure C.8 XML Schema of Context Query Message

Figure C.9 XML Schema of Context Response Message

Figure C.10 XML Schema of QueryRequestMessage

Figure C.11 XML Schema of QueryResponseMessage
Figure C.12 XML Schema of Peer Monitor Query Message

Figure C.13 XML Schema of Peer Monitor Response Message
Appendix D  Questionnaire

☑ Please tick the box.

**Part 1: Effectiveness and accuracy**

1. Relevant News items are located at the top of the ordered list.
   - [ ] Strongly disagree    - [ ] Disagree    - [ ] Neither agree nor disagree    - [ ] Agree    - [ ] Strongly agree

2. PPNews learn your interests quickly.
   - [ ] Strongly disagree    - [ ] Disagree    - [ ] Neither agree nor disagree    - [ ] Agree    - [ ] Strongly agree

3. Overall, PPNews system filtered News items are effective and accurate.
   - [ ] Strongly disagree    - [ ] Disagree    - [ ] Neither agree nor disagree    - [ ] Agree    - [ ] Strongly agree

**Part 2: Efficiency**

4. PPNews response time is satisfactory.
   - [ ] Strongly disagree    - [ ] Disagree    - [ ] Neither agree nor disagree    - [ ] Agree    - [ ] Strongly agree

5. PPNews are able to reduce your keystrokes.
   - [ ] Strongly disagree    - [ ] Disagree    - [ ] Neither agree nor disagree    - [ ] Agree    - [ ] Strongly agree

**Part 3: Usability and layout**

6. PPNews application can install and use easily.
   - [ ] Strongly disagree    - [ ] Disagree    - [ ] Neither agree nor disagree    - [ ] Agree    - [ ] Strongly agree

7. You are satisfied by PPNews proactive pushing features.
   - [ ] Strongly disagree    - [ ] Disagree    - [ ] Neither agree nor disagree    - [ ] Agree    - [ ] Strongly agree

8. The layout of the PPNews system is appropriate.
   - [ ] Strongly disagree    - [ ] Disagree    - [ ] Neither agree nor disagree    - [ ] Agree    - [ ] Strongly agree
9. Overall the PPNews system is easy to use.

☐ Strongly disagree ☐ Disagree ☐ Neither agree nor disagree ☐ Agree ☐ Strongly agree

Part 4: Connectivity, stability and reliability

10. The PPNews System is stable with no crashes.

☐ Strongly disagree ☐ Disagree ☐ Neither agree nor disagree ☐ Agree ☐ Strongly agree

11. You have not notice any disconnection/downtimes during usage.

☐ Strongly disagree ☐ Disagree ☐ Neither agree nor disagree ☐ Agree ☐ Strongly agree

12. PPNews system is overall reliable.

☐ Strongly disagree ☐ Disagree ☐ Neither agree nor disagree ☐ Agree ☐ Strongly agree

Part 5: Comments

13. Please describe any other comments.

____________________________________________________________________
____________________________________________________________________
____________________________________________________________________
____________________________________________________________________