3-25-2009

Development of a New Client-Server Architecture for Context Aware Mobile Computing

Feng Gui
Florida International University

DOI: 10.25148/etd.FI10022519

FLORIDA INTERNATIONAL UNIVERSITY

Miami, Florida

DEVELOPMENT OF A NEW CLIENT-SERVER ARCHITECTURE FOR CONTEXT AWARE MOBILE COMPUTING

A dissertation submitted in partial fulfillment of the

requirements for the degree of

DOCTOR OF PHILOSOPHY

in

ELECTRICAL ENGINEERING

by

Feng Gui

2009

To: Dean Amir Mirmiran
College of Engineering and Computing

This dissertation, written by Feng Gui, and entitled Development of a New Client-Server Architecture for Context-Aware Mobile Computing, having been approved in respect to style and intellectual content, is referred to you for judgment.

We have read this dissertation and recommend that it be approved.

Armando Barreto

Naphtali Rishe

Jean H. Andrian

Malek Adjouadi, Major Professor

Date of Defense: March 25, 2009

The dissertation of Feng Gui is approved.

Dean Amir Mirmiran
College of Engineering and Computing

Dean George Walker
University Graduate School

Florida International University, 2009

DEDICATION

I dedicate this dissertation to Jiong Zhao, my wife, and my parents. Without their support and sacrifice, I would not have possibly achieved such milestone in my life. I would like to thank my professor Dr. Malek Adjouadi who broadened my research interests and advised me throughout my dissertation work. I would like to express my gratitude and thanks to my committee members, Dr. Jean Andrian, Dr. Tadeusz Babij, Dr. Armando Barreto, Dr. Naphtali Rishe, and Dr. Wunnava Subbarao, for all the help they provided and all the recommendations they made. I want to express my gratitude to the CATE lab members for their support and encouragement during the preparation of this dissertation, especially to Mr. Magno Guillen, Ms. Xiaozhen You and Mr. Melvin Ayala.

ACKNOWLEDGMENTS

The research infrastructure of the CATE center and the financial support provided by the National Science Foundation through grants HRD-0317692, CNS-0426125, CNS-0520811, CNS-0540592 and IIS-0308155 are greatly appreciated.

ABSTRACT OF THE DISSERTATION

DEVELOPMENT OF A NEW CLIENT-SERVER ARCHITECTURE FOR CONTEXT-AWARE MOBILE COMPUTING

by

Feng Gui

Florida International University, 2009

Miami, Florida

Professor Malek Adjouadi, Major Professor

This dissertation studies the context-aware application with its proposed algorithms at client side. The required context-aware infrastructure is discussed in depth to illustrate that such an infrastructure collects the mobile user's context information, registers service providers, derives mobile user's current context, distributes user context among context-aware applications, and provides tailored services. The approach proposed tries to strike a balance between the context server and mobile devices. The context acquisition is centralized at the server to ensure the reusability of context information among mobile devices, while context reasoning remains at the application level. Hence, a centralized context acquisition and distributed context reasoning are viewed as a better solution overall.

The context-aware search application is designed and implemented at the server side. A new algorithm is proposed to take into consideration the user context profiles. By promoting feedback on the dynamics of the system, any prior user selection is now saved for further analysis such that it may contribute to help the results of a subsequent search.

On the basis of these developments at the server side, various solutions are consequently provided at the client side. A proxy software-based component is set up for the purpose of data collection. This research endorses the belief that the proxy at the client side should contain the context reasoning component. Implementation of such a component provides credence to this belief in that the context applications are able to derive the user context profiles. Furthermore, a context cache scheme is implemented to manage the cache on the client device in order to minimize processing requirements and other resources (bandwidth, CPU cycle, power). Java and MySQL platforms are used to implement the proposed architecture and to test scenarios derived from user's daily activities.

To meet the practical demands required of a testing environment without the impositions of a heavy cost for establishing such a comprehensive infrastructure, a software simulation using a free Yahoo search API is provided as a means to evaluate the effectiveness of the design approach in a most realistic way. The integration of Yahoo search engine into the context-aware architecture design proves how context aware application can meet user demands for tailored services and products in and around the user's environment. The test results show that the overall design is highly effective, providing new features and enriching the mobile user's experience through a broad scope of potential applications.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER I

INTRODUCTION

With the technological advances in wireless communication, mobile devices such as cellular phones evolved from single-task gadgets, for example conversation only, to multi-media devices in mobility. The Internet Protocol (IP) convergence concept in recent years accelerated the trend for ubiquitous computing, playing henceforth an increasingly important role in our daily lives. The concept of context awareness originally stems from the research of human-computer interaction (HCI). One of the objectives in HCI research is to design computer interfaces that reduce user intervention while providing more relevant functions/services to the user. Context awareness came across the attention of HCI research community as the "hot topic" a few years ago. One group of researchers tried to standardize the context definition in a hope that a set of well-defined context concepts is able to interpret mobile user's changing situation. However, some researchers argue that these predefined context notions are still too vague or general to understand the changing context. Due to mobility, users significantly expand their personal space far beyond fixed indoor environment such as office and home.

At the present time, mobile users are very creative in making use of mobile devices to affect almost every social aspect. Obviously, a fixed set of context notions has very limited success in learning the mobile user's intention under various scenarios. Another group of researchers point out that context awareness is a study that must also incorporate mobile users' psychological behavior, group dynamics, and social interaction. This group of researchers laid their framework on the observations of societal patterns and common psychological behavior of the mobile users within a town or city. This is clearly a challenging and broadly defined problem. In this dissertation, the mobile context is examined and studied focusing on the user-centered empirical approach. This approach is an attempt at exploiting the information environment to suit specific user needs. Context awareness therefore can be viewed as a facilitator for mobile users who seek reduced human interaction and an improved machine-learning process.

With this undertaking, the main objective of this dissertation is to develop a novel context-aware search application which can provide data and services relevant to the user's given situation and whose design is able to adapt to the day to day changes and user dynamics. The programming language considered throughout this dissertation is Java due to its portable nature, with the notion "Write once, run anywhere". Three integrated development environment (IDE) tools, Netbeans, Sun Java Wireless Toolkit, and Eclipse, are utilized to implement simulation code. The Java version is 1.6 which includes the IDE Netbeans 6.1.

In Chapter II, the definition of context is provided. Context research remains one of the most important topics in the design of human-computer interfaces (HCI) as well as in applications of artificial intelligence (AI). Ubiquitous computing is still a relatively young discipline. Researchers with different backgrounds in HCI, mobile computing, computer vision, and distributed systems, to name a few, contribute to this research field. A brief history of mobile context is thus given as a background. Research on mobile context, shifting from the traditional fixed indoor patterns, looks into aspects of psychological and social behavior as well, making this research problem even more challenging. In particular, the context entities are identified to describe the state of the mobile user in a given situation. User profiles which consist of context entities are compiled at the different distributed systems and mobile devices. The context profiles are utilized to describe mobile user's current state. Context data modeling deals with raw context data acquired from various data sources including wireless sensor network and mobile devices. Context data can be discrete or continuous. In addition, the readings vary widely in format. Context modeling converts data into machine-readable formats. A brief review of context models is given. Over the years, different models of ubiquitous computing were proposed. The most common model is the one based on the conceptual view which mimics the way human perceive the world. The building block of the conceptual modeling is the context entities. Because the high-level context abstraction is distributed to the mobile device, the context modeling does not include this type of abstraction. Key-value pair and object-oriented modeling are the most used structures. However, they are ad-hoc and application specific. XML is used to describe the context data. XML is widely accepted to express data and semantics. The advantage of XML is that a lot of parsers exist in different programming languages. This gives the application developer freedom over the developing tools. The issues of context computing are discussed and possible solutions are introduced in this chapter. These solutions lead to many useful applications. Some interesting applications are enumerated to see how context-aware devices help out users.

Chapter III starts with an introduction of the context-aware architecture. Advances in both mobile devices and carrier network have led to the context aware utility and its preponderant use in day to day activities. The introductory section looks into some of the existing solutions. Each solution has its own advantages and disadvantages, requiring different context architectures with varying hardware and software demands. However, network carriers are reluctant to deploy sensor network due to high cost and lack of high-visibility application. Consequently, a low cost sensor network is provided to address this difficulty. A wireless sensor network takes advantage of the latest WiFi technology, Bluetooth, and mesh network. The required context-aware infrastructure consists of context interpreter, service registry, context manager, request interpreter, policy registry, and context-aware search manger. The context-aware search manager takes user context profiles into account. The user context files are compiled at mobile devices taking advantage of many applications and the distributed computing power. Context profiles are further processed and augmented with other user information at context server. Thus, the search results are tailored to user context. The main objective of context server is to distribute the user context data among applications. Hence, context reusability shields application developers from low level context acquisition. Unlike existing architectures, the context reasoning is distributed to mobile devices.

Chapter IV introduces the algorithms that have been developed at the client side to capture and extract the user context as a foundation that supports the design of the cache scheme, the search engine and the required context-aware server. The relevance of this is that context profiles (user profile, device profile, environment profile and data profile) need to be fed to the context-aware server, which in turn, will supply all the different context aware applications. Within this design concept, the context aware application developer can now focus more on the context-aware application itself rather dealing with low-level hardware which acquires the context data. These algorithms include: (1) Virtual frame and reference frame calculation; (2) context-aware cache scheme. Based on these technology breakthroughs, a client proxy is designed to manage the context computing on the mobile devices. Identifying context entities, computing context weights, and compiling context profiles are the major tasks conducted by the proxy. Context weights are important measurement of the relevance in user context. The context abstract is quantified and digitized. The values of these context weights should reflect the significant changes in the given situation. The context profiles contain all the recognized context weights. Context profiles are classified into different categories. Device profile describes the hardware configuration of the mobile device and software installation. The user profile reflects mobile user's personal characteristics such as gender, age, and hobbies. Likewise, the environment profile contains information regarding weather, noise level, surrounding social activities, and available resources. The client proxy compiles these profiles and relieves the operator's network of number crunching on context profiles. The context profiles are updated either periodically or on the change of major context entities. The proxy sends the compiled context profiles to the server such that the network (server) is aware of the mobile user's given situation.

Chapter V provides all the simulations based on real-world scenarios to evaluate the merits of (1) the context search as compared to web-based search (Yahoo and Google) using different roaming scenarios, (2) the use of the cache algorithm in terms of virtual and reference frames which are used to capture user context based on the same scenarios in (1). These simulations prove the effectiveness of the context-aware search application built upon the required context aware architecture. Java applications are written to simulate the developed software modules at different levels. The database server is MySQL which is very popular among open source community. The MySQL server version is 5.1.30. A relational database is designed to store user context information and service provider's information. The database also supports the context search algorithm. A few assumptions are made due to lack of resource for hardware infrastructure. For these simulations, the user information is assumed to be provided from the required context-aware infrastructure which may include wireless sensor networks, cellular networks, and mobile user devices together with a context aware server. It is noted that all the user information is further converted into digital format in order to extract and process the user context. The simulation takes the required context information from the text files as if the data were provided from the context aware server. Furthermore, test scenarios are generated based on average user activities. Test data and results are presented accordingly. In addition, Yahoo Search API is used to get preliminary search results. These preliminary results are then ranked based on the user context. Furthermore, any user selection will be saved for future reference as a feedback mechanism.

Chapter VI concludes with a retrospective on the accomplishments of this dissertation and provides a reflection on key technological developments that would advance this research field within feasible and economical bounds.

CHAPTER II

A RETROSPECTIVE ON CONTEXT AWARENESS

2.1 Notion of Context

Gordon E. Moore, co-founder of Intel, predicted in 1965 that the number of transistors on a chip would double about every two years. His prediction is what is known as Moore's Law today [1]. In this line of thought however, the size of the computing device changed with the changing application(s). To this day, when massive computing power is involved, it would be because of the inordinate amount of data it needs to process and it would most certainly be shared by many users/researchers. In present days however, the tendency is more for individuals to use multiple personal computers and or mobile, for example notebook and desktop computer, at home and office. This tendency is paralleled in wireless communication through the use of notebooks, PDAs, cellular phones, and other mobile devices with a high proliferation rate [2].

These mobile devices are able to integrate more functions such as MP3 music, digital camera, video stream, mobile TV, Web access, to name a few. Texas Instruments, leading cellular phone chip supplier, offers OMAP^{TM} family of multimedia application processors which integrate the above applications in single-chip solution [3]. Wireless carriers, on the other hand, continue to improve the network infrastructure as new protocols and technologies emerge. Japanese wireless carrier, NTT DoCoMo, became the first carrier in the world to launch the commercial 3G wireless network in 2001. As of

December 2006, the 3G wireless users in Japan alone reached the 60 million mark (http://www.iresearch.com.cn). It is estimated that the number of wireless users in the world will surpass 4 billion by 2010. That is why the widespread of wireless devices is now leading the way in the new research field of ubiquitous computing.

A decade ago pioneer Mark Weiser envisioned that a human would live in an environment surrounded by hundreds of invisible computers connected with wireless networks [4]. Indeed, the ubiquitous computing is now interwoven into our daily lives such that mobile users could access information anytime and anywhere without even knowing the aid of these computers set up around them. In order to achieve Weiser's vision of ubiquitous computing, understanding user context becomes imperative because such wireless network must be sensitive to the user's context data which are not explicitly given to these invisible computers in mobile computing. Ubiquitous computing requires the invisible computer networks to cope with context awareness at different levels. The intelligent mobile devices must be aware of the CPU, memory, Operating System, and other device configuration of surrounding computers at the hardware level. At the network level, variations in topology, bandwidth, protocols, congestions, and other real-time traffic parameters must be considered by user devices. At middleware level, the context information affects the application performance, resource allocation, and execution optimization. At application level, the applications must adapt to the interactions, network variation, and changing inputs due to the mobile environment [5]. Finally, the invisible systems must deal with issues such as output display at user interface level. Obviously, such context-aware wireless networks must actively acquire, analyze, and adapt to mobile user's given context such as physical environment, social activities, and other dynamic characteristics at different levels without consuming much of the user's attention.

What is the mobile context then? Context is defined as "the interrelated conditions in which something exists or occurs" in Merriam-Webster's dictionary. Wikipedia defines that "In mobile computing, context is the circumstances under which a device is being used, e.g. the current occupation of the user." The definitions of context refer to the understanding of the "interrelated conditions" and "circumstances" of the mobile user. The context must reflect the artifacts and environment the mobile user interacts with. Researchers believe that there exists a strong relationship between context and entities in the environment [6, 7].

So what exactly describes the "interrelated conditions" and "circumstances"? Ideally, context should fully describe the state of the mobile users and their surrounding world at a given time. In general, it is not easy to single out the mobile user's context from objects and entities which are not relevant to users' intention. After all, too many things or entities around us have context. Humans have different knowledge, expectations, experiences, priorities, social roles, and tasks to view the world surrounding them. Hence, individuals select the objects in the real world in a subjective way and describe context of the world complexity in an unstructured fashion. Furthermore, the available sensors and sensing technologies limit us to learn the real world complexity in technical aspect. For example, the video cameras can tape mobile users and their surrounding scenes.

However, human interpretation is still needed to describe the mobile users' emotional expression such as anger and laugh. Based on the above observation, researchers try to refine the definitions of context in ubiquitous computing. Dey defines context as "any information that can be used to characterize the situation of an entity. An entity is a person, place, or object that is considered relevant to the interaction between a user and an application, including the user and applications themselves [8]. Chen and Kots state that two aspects of mobile context must be addressed. Firstly, the behavior of applications on mobile device should reflect characteristics of the surrounding environment. Secondly, the mobile applications themselves indicate the user's context in some cases, if not necessary in all. They concluded that "Context is the set of environmental states and settings either determines an application's behavior or in which an application event occurs and is interesting to the user" [9]. The context in ubiquitous computing must reflect the mobile user's current state including physical and psychological behavior, the objects embedded with computing and communication capabilities, the mobile environment and scenarios, and mobile application's interactions based on the explicit and implicit user input.

In contrast to prevailing viewpoint on definition of context, Rehman challenges the research community with his notion. He argues that the objects in the real world do not have context at all even though they might have physical properties. He gave an example on this point. A pair of computer loudspeakers on the desk does not have any context in his point of view. Instead, the real world objects form part of the context in interaction between the user and applications [10][11].

2.2 Context Entities

Having defined mobile context, context entities can now be identified in more details. In general, context entity refers to person, location, time of the day, event, objects, artifacts, or combination of the above that are related to the interaction between the user and mobile devices. User's age, gender, weight, height, and other personal data become the context entities that reflect the user context. Similarly, temperature, noise level, lighting, and location are the context entities that describe the environment context. Context entities can be classified into the following main categories;

- Device context entities refer to the hardware configuration and software applications of the mobile devices. Antenna, LCD display, keypad, stylus, Bluetooth earphone and so on are all good examples of context entities. On the other hand, Operating Systems, browsers, address books are mobile software applications. The middle wares which are not so obvious to mobile users are also device entities.
- User context entities include information regarding the specific mobile user. Age, gender, height, weight and other personal information fall into this group. In addition, user personal preferences such as hobbies, settings for applications, and browser bookmarks are also considered as entities which can be used to derive user context.

- Environment context entities consist of information of the surrounding area of the mobile user. The weather, temperature, lighting, noise level, location, near-by activities, and people are all considered the environment context entities.
- Time is the arguable context entity. Some researchers consider the time context is included in the user activities. Other researchers argue the time is important context which deserves more attention in user context analysis, because it reveals the temporal aspects of the context entities. This dissertation supports the viewpoint of the later group.

However, the above categories do not serve the purpose of defining all context entities to cope with the dynamic and unpredictable mobile user's characteristics. Instead, these categories serve to list the entities that are most commonly known to ubiquitous computing. Some context entities that are not so obvious sometimes play an important role in some mobile applications. For instance, the history context of the mobile user is crucial to understand if the user's context remains little changed over a period of time. There are many context entities to be addressed within a context aware architecture beyond the traditional approach, which limits context awareness to location awareness [11]. In this dissertation, the empirical and user-centered approach is applied as means to study the mobile user's context.

A variety of sensors are applied to capture the context entities in different categories. The details of context acquisition are specified in the later chapter. The industry trend in the past saw the decoupling of sensor layer and the application layer. The purpose of decoupling is to remove duplication in hardware logic and shield application developers from low-level hardware details [12]. Context-aware applications in general share the context entities acquired from the sensor layer or from the wireless networks. Thus, modeling these context entities becomes an issue. In particular, how can these context entities be represented? Some context entities such as age, weight, and noise level are easy to present and disseminate in digits. While other context entities are not so straight forward. For example, activities of the surrounding area are certainly of interest of many applications such as tour guide systems. Modeling context entities will be discussed later.

2.3 Fundamental Concepts of Mobile User Context Profiles

After acquiring/extracting the context entities from the user context, the next step is to organize the context entities into context profiles. Context entities are grouped into these files in terms of context nature. In this research, context entities are classified, but not limited, into the following major categories;

- User profile describes the user's characteristics, preferences, activities, emotions, and so on. For example, user's age, gender, height, weight, health conditions, past incidents, and planned events can be sorted into this category.
- Device profile describes the configuration and functions of the hardware. In addition, this profile includes the software applications and user interfaces. Examples of context entities in this profile are processor, memory, display type, device size, and operating system.

- Environment profile describes surrounding area of the mobile user. This profile stores user location, weather, noise level, temperature, etc.
- Data profile caches user's data in the local memory.

The context profiles facilitate applications in the upper layer. Applications in the upper layer no longer have integrated sensors, so they share context profiles. At this point, the concepts of context, context entities, and context profiles are all introduced. Figure 2.1 illustrates the bottom-up modeling proposed in this dissertation.

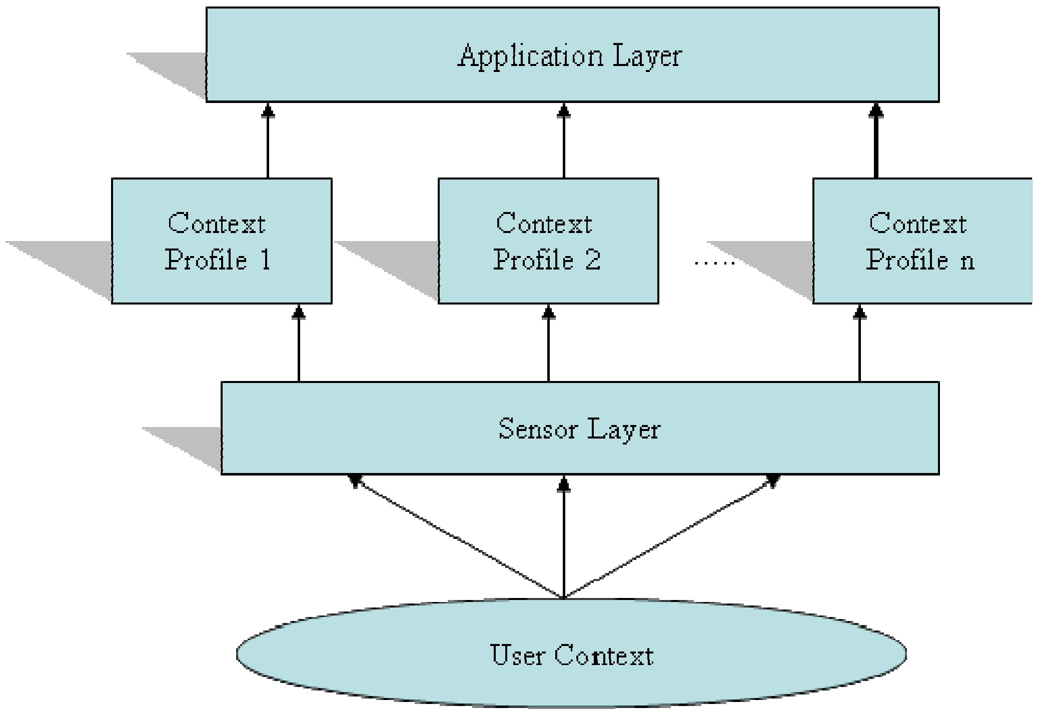


Figure 2.1 Bottom-up Approach for Context Modeling

2.4 Context-Aware Mobile Computing

A decade ago Mark Weiser predicted that future environment would be surrounded or embedded by "invisible computers" connected with networks. In his article, he states "the most profound technologies are those that disappear. They weave themselves into the fabric of everyday life until they are indistinguishable from it" [4]. These invisible computers comprise the context-aware systems which complete tasks related to the roaming users as if the nomadic users did them by themselves. Context-aware computing is a paradigm for developing smart systems that can derive mobile user's need and adapt accordingly on the user's behalf. Wikipedia explains, "In computer science it [context awareness] refers to the idea that computers can both sense, and react based on their environment. Devices may have information about the circumstances under which they are able to operate and based on rules, or an intelligent stimulus, react accordingly." Ryan defines context awareness as a term "that describes the ability of the computer to sense and act upon information about its environment, such as location, time, temperature, or user identity. This type information can be used not only to tag information as it is collected in the field, but also to enable selective responses such as triggering alarms or retrieving information relevant to the task at hand [13].

Schilit characterized context awareness as follows "such context-aware software adapts according to the location of use, the collection of nearby people, hosts, and accessible devices, as well as to changes to such things over time. A system with these capabilities can examine the computing environment and react to changes to the environment" [14]. Unlike the idea of invisible computers, Rehman recently points out that context-aware system is based on the interaction between the mobile users and system. He argues that user should play an active role to understand what assumptions the context-aware systems make inference on user's behalf. The interactive systems, in his mind, should take the user's feedback. Furthermore, the interactive model allows user to evaluate the context-aware systems. Over the years, context-aware computing became an interdisciplinary research that involves embedded systems, distributed system, artificial intelligence, human computer interaction, architecture design, and computing science. Researchers with different backgrounds such as mobile computing, AI, distributed systems, and HCI have contributed in all aspects of this relatively young field.

The European Commission proposed project which echoes Weiser's invisible computer, "A vision of the future is one in which our world of everyday objects and places becomes infused and augmented with information processing and exchange. In this vision, the technology providing these capabilities is unobtrusively merged with real world objects and places, so that in a sense it disappears into the background, taking on a role more similar to electricity an invisible pervasive medium" [15]. To realize Mark Weiser's ubiquitous computing, it implies that user's interaction for input should be minimized, if not possible to eliminate the user interaction all together. The "invisible computers", in Mark Weiser's vision, do not mean the physical size of the computers is too small to be visible. Nor does it mean that using portable devices, as user goes, is a good example of ubiquitous computing. Instead it requires that the user interface and human-machine interaction in ubiquitous computing are totally different to the traditional wired computers, since keyboards and displays are no longer the only interaction between the mobile user and computers. The mobile devices typically have reduced LCD display and tiny buttons. In ubiquitous computing, the artifacts and environment that surround the mobile user are included as part of the interaction for the context-aware systems. The inputs from the mobile users are implicitly acquired by the context-aware systems as much as possible. This process is achieved by the collaboration of mobile devices and computer networks around the user. In a sense, the context-aware computing is aimed to assist or even enhance the user interface of the mobile devices.

The invisible computer networks have to be aware of the user’s situation including social and psychological behaviors. It indicates that such context-aware system is able to distribute the user’s context profiles wherever the user goes. This feature overcomes the asymmetry of the invisible context-aware system. In other words, it is not feasible to build context-aware networks with the uniformly distributed hardware and topology all over the areas. Wireless operators are often bound to limited resources. In economic sense, they would focus on the efficiency of the network deployment and invest heavily to areas that generate the most profit. Even within the same operator, it is common that different wireless technologies can be deployed simultaneously. For example, US wireless operator, Sprint, is well on the way to deploy the first WiMax network across the nation which is meant to supplement the 3G CDMA network. Likewise, the Chinese operator, Unicom, has deployed both GSM and CDMA networks across China. The co-existence of various wireless technologies challenges the architecture design, user context acquirement, and context distribution.

In the early days of context awareness research, some experts debated what user context comprises. These same experts, inspired by the study of traditional human computer interaction, proposed to predefine a set of context entities which would cope with all user situations. However, the mobile users significantly expand their personal space far beyond fixed indoor environment such as office and home. In addition, people utilize their mobile devices creatively in their personal activities or social events. Nowadays, mobile devices find their way to penetrate our daily lives as mobile users utilize them for conversation, email, games, TV, ecommerce, banking, video stream, and so on. The scope of the mobile device usage is thus extended to cope with the various aspects of our society.

Therefore, some researchers argue that the all-embracing set of context entities is too general to capture the user's context in particular situations. Instead, this group of researchers proposed an empirical approach to find the regularities persistent within particular situations. For example, Luff and Heather [16] are the pioneers who explored how the mobility of artifacts impacts the traditional working settings. In the clinics they studied the "micro-mobility" of medical records. They revealed how the mobility of traditional medical records brings collaboration among doctors and patients. The introduction of computer based medical records changes the way the doctors interact with patient. However, the traditional medical records remain the important resource within medical practice, because it preserves the "ecological dexterity" for the doctors. Simply put, traditional medical records assist the doctor with the communicative flexibility. At the construction site, the specially designed mobile devices are distributed to foreman in order to assist them in the daily operation. Contrary to what application designer hoped for, the mobile devices which are supposed to be carried by foreman remain in the fixed office setting and are managed by the office clerk. The empirical studies contributed by Luff and Heather indicate context-aware application design must be socially acceptable and useful. The design of context-aware applications must assist the existing collaboration instead of overtaking it in a dramatic fashion. Tamminen and Oulasvirta carried out experiment on 25 adults in Helsinki to study the rapid change of contexts in everyday urban navigation [17]. Their study analyzed the impact of the social interaction, psychological behavior, and group dynamics toward user context. The experiments helped the system designer to better understand the hidden factors within user context along with tangible context entities in the environment. Understandably, the important factors and entities in one situation might not be so crucial in other situations. Even same entities have different impact from one setting to another. The design of context-aware systems can thus be fine-tuned out of trials and errors based on the experiments. More importantly, the empirical studies also provide an accurate evaluation to measure the effectiveness of context-aware design.

2.5 Issues of Context Awareness

The essential part of the context awareness is to design an artificial intelligent architecture which acquires, analyzes, learns, models, and distributes the mobile user's context [18]. The mobile devices can also contribute to undertake some of the context server's functions in terms of context acquisition and analysis, as more advances are made in this field of research and as new challenges are ultimately revealed.

Acquiring the user information is the first step towards understanding the changing context. The hardware infrastructure must be provided to collect user context. The solution is two-pronged:

- On the one hand, technological advances enable hardware manufacturers to integrate more hardware logic into wireless terminals. For example, the iPhone from Apple recognizes user's face and dims the LCD display if user's face touches the iPhone during the conversation. Korean cell phone maker allows the user to lock/unlock the cell phone with finger prints. These examples show the trend that mobile devices are likely to play a more active role to acquire the user's context.
- On the other hand, wireless operators have little progress in deploying infrastructure due to the cost factor. Even though the 3G networks can be used to disseminate user context information, most 3G technologies do not take context acquisition into account. Extra hardware infrastructure, especially sensors, must be put in place additionally. The major barriers that hold operator's infrastructure deployment are high cost and lack of the so-called "killer" applications.

Furthermore, one is also called to reflect on how context acquiring processes at mobile devices and carrier's network are being coordinated? Once the context data are acquired, we still can ask: how can they be represented? Context data can have all sorts of readings in continuous and discrete forms. How can context data be transformed in a machine readable structure? Context modeling is viewed in this dissertation as the answer for this question. Context modeling is a process to convert context data into a structured format. In the past, researchers proposed some ad-hoc data structures to express and exchange context information. Object oriented approach was one of the popular data structures to model context entities, as context entities were treated as objects. This approach originates from the programming aspect obviously. XML seems a more interesting way to distribute the context entities, because XML is widely accepted and open to users. Key-value pair is also proposed as a solution due to its simplicity in modeling the context entities.

Another issue is how mobile devices share user context if there is no such infrastructure available to support such context reusability [19]. Researchers proposed solutions ranging from context middle ware to context server in order to support reusability [20]. Nowadays, mobile users are no longer limited to one mobile device anymore. Some mobile users carry multiple mobile devices. A truck driver might drive a truck which has a wireless GPS device in addition to a cell phone. A delivery staff might be equipped with cell phone and wireless devices that scan and update schedule remotely with the company's database.

These mobile devices might subscribe to different networks. There are underlying variations in the individual networks. For instance, WiMax is an emerging technology which is likely to coexist with other 3G technologies. It is clear that the context sharing infrastructure must interact with heterogeneous networks. For example, Henning Maab proposed a centralized location-information server which is integrated in the network platform [21]. This centralized architecture design is based on directory service. It provides an application programming interface (API) for mobile devices to query location. Furthermore, different mobile devices need different context information. A mobile scanner designed for a specific task does not process more user information than what it is designed for. For instance, the noise level in the surrounding environment has no value for the task-specific mobile scanner. Clearly, the context sharing mechanism must allow the applications to request related context information.

Despite the noted progress, there are few hardware limitations facing the notion of context awareness. The batteries on the mobile devices restrain the usage of mobile computing. Users demand more applications that consume more power. For instance, the fully charged battery can only support a few hours of conversation on iPhone. Users who are going to watch the mobile TV are likely to shorten usage time before next recharge. In addition, the average memory capacity on a cell phone is less than 1 GB. The typical LCD display on the cell phone is somewhere between 1.8 inches to 2.5 inches. The context aware tools/systems must be running in the background as much as possible. Mobile applications that users work on should have the priority to run in the front. User's applications should not be interrupted at all. Therefore, the context aware systems are required to use as little system resource as possible. Also the hardware variation in mobile devices makes the design more complicated.

Furthermore, if the surrounding artifacts and environment are included as part of the interaction for context-aware systems, user's personal data will be shared with computer networks explicitly and implicitly. This is clearly an issue of privacy. How comfortable are mobile users if their information is distributed to various computer networks along the way they travel? It is very likely that a policy or protocol must be in place to guard data safety. The data protection requires a mechanism to ensure user's context to trustworthy computer networks. The privacy issue, no doubt, adds complexity to the design of context-aware systems. If mobile users wish to turn off the context awareness feature, they must be provided a means to do so.

2.6 Context Modeling

The most important feature of a context-aware architecture is the reusability of context among applications such that the application developers can focus on the high level application development without the drag on context data acquisition. Because of the heterogeneous nature in context data acquired from various types of sensors, it is crucial to represent context data in a structured format. Put simply, it is desirable to transform the context entities and user space into a digital world which is suitable for machine processing. Context modeling plays a role of converting, analyzing, and composing context into structured data format which can be stored and retrieved for context-aware applications.

Data structure is one relevant domain in context modeling. In the early years of context-awareness research, location-based context modeling revolved around the location. Geometric model and symbolic model are two modeling structures which are ad-hoc in nature. Today, there is still no standard toward the data structure in context modeling. The most common data structure is the so-called ad-hoc key-value pairs. In this structure, entity is defined as a key and its value is associated with the context key. This data structure is simple and it works. The object-oriented model is a much more balanced modeling structure. This is so because entities can be perceived as objects. Entity types, attributes, context values, and relationships can be mapped into objects, object variables, and object methods. The object-oriented model is also an ad-hoc approach. It abstracts context in high level and encapsulates context data processing in low level. Object-oriented model facilitates context reusability and aggregation. However, some researchers criticize key-value pair and object-oriented structures because they believe these ad-hoc structures lack expressiveness and extensibility. In addition to key-value pair and objected-oriented structure, developers also make use of database to store and retrieve context data. The advantage for database is that standard query is used to facilitate applications using different programming languages. Retrieving from database is faster with database server. Critics point out that database server is not able to resolve context inconsistence.

Over the years, researchers developed context models from different aspects of context awareness. Some models, for example user interaction model, which focus on human-computer interfaces. Conceptual models are based on the context entities. Policy driven models adapt to protocols, network parameters, and carrier policies. The section dwells on some of the context models.

2.6.1 Modeling Languages

In general, various tools exist to model the context entities and relationships among them. Modeling languages are used to describe the context entities and the relationship among them. One of the most used languages is UML (Unified Modeling Language). UML has been widely used in academia due to its clear semantics and graphical presentation. The advantage of the UML is that object classes are fit to model the context entities, attributes, behavior, and relationships. In addition UML links are utilized to associate interaction between context entities. The generated UML charts which include the UML classes and links describe structured context data in the high level forms without the implementation details. Also the UML flow charts demonstrate the events or services triggered when the context data changes between the mobile users and service providers. The other modeling language is ontology language. Context entities in the mobile environment can be entered and deleted from the ontology database in the runtime. Raw context entities can be composed to composite context data with the ontology language constructs like subClassOf. While the disjointWith constructs describe relationship among context entities. One advantage of ontology language is that researchers can use it to build the reasoning mechanism and define the application's behavior upon the change in context.

2.6.2 User Interaction Modeling

This type of modeling is based on the user interactions. Variations exist in how interactions are defined and the scope of interactions, but the main categories of interactions remain the same. The main user interactions are user-application interactions, user-network interactions, application-network interactions, and interactions among users. Entities, such as user preferences, network traffic, bandwidth, application requirements, become part of the context only when the interactions happen. Unlike user-center modeling, the focus in this modeling is on the interactions. The user-application interactions include interactions between user and interfaces of applications. User input through key pads, voice recognized module, touchable screen, and stylus are typical interactions. Application behaviors or feedback are interactions controlled by applications.

The user-network interactions cover the user's behavior toward network variation and network management of user requests from mobile applications. In general, mobile users respond to network variations in terms of bandwidth, connectivity, congestion, and usage cost. For instance, if the user roams out of Wi-Fi network, the underlying cellular network overtakes the connection from vertical hand-off. However, the bandwidth is most likely reduced. So the user might suffer quality of service (QoS) for video streaming applications via network. With low bandwidth or network congestion, the user might turn off the application. On the other hand, the modern network improves in handling user's traffic. The network is able to classify user traffic based on issues such as QoS. Hence, the network adapts to user device and applications.

The application-network interactions refer to mutual adaptation between the user applications and networks. For example, the IPV6 has priority field for IP packets which allow the network to adapt and allocate network bandwidth accordingly. Thus QoS can be improved over the IPV4. Human interactions are indirectly involved in this type of interactions.

2.6.3 Activity-Oriented Context Modeling

Unlike the user interaction modeling which centers on the interactions among user, machine, network, and application interfaces, the activity-oriented context modeling is based on the external resources and internal mental process. The external factors are defined as human, artifacts, and environment. The internal mental process refers to user goals and beliefs. The user activities consist of both external factors and internal mental states. In essence, this model requires the designer to predefine the activity plans which also include all the social context factors and cognitive contexts as parameters. The context adaptation is managed by monitoring the switch among user activities. The advantage of such modeling is that system design is simple, since the system just needs to follow the context “template”. The weakness of such modeling is that it does not scale well as more predefined activities are under monitor. In addition, it cannot deal with activities that are not predefined. Finally, it is very difficult if not impossible to detect user’s internal goals and beliefs.

2.6.4 Policy Driven Model

The policy driven modeling is quite different from other models where human-computer interfaces are emphasized or user-centric context take center stage. This type of modeling makes services adapt to different communication protocols and network characteristics. In the upcoming years, service providers must offer mobile user services which utilize the Next Generation Networks (NGN) which consist of emerging heterogeneous networks, wired networks, and various wireless networks. This requirement makes service context aware to the underlying networks. The policy-based network management has become the subject of extensive research in the past decade. Network administrators are relieved from configuring individual devices to operate under heterogeneous networks. Because services are context-aware and operating on the context aware architecture, so network administrators only need change the policies governing the networks. The policy modeling takes the context-aware object oriented approach which is, in general, based on the Policy Core Information Model (PCIM). Network variables and management policies are mapped in to modeling objects.

2.6.5 Peer-to-Peer Light-Weight Component Model

This model is typically for small-scale applications. The idea of Peer-to-Peer light-weight component model is to develop containers of components (middleware) which abstract and encapsulate the communication protocols utilized to interact among mobile applications. The increasing complexity in software and hardware quickly dampens the development of flexible and dynamic applications. Especially, developers of mobile applications need to take advantage of emerging communication protocols. The peer-to-peer component model shields the protocol details from the application developers such that developers can focus on the features or functions their applications provide to mobile users. The modeled component is a collection of objects which can provide services to remote peers. The motivation for these components is to provide infrastructural support for context-awareness in mobile peer-to-peer applications. The interfaces of these components are unified such that the details of underlying communication channels are taken care of by these containers. The container has management objects which adapts to user's mobile devices by selecting appropriate protocols. The peer-to-peer applications are installed on resource-constrained mobile devices and offer seamless configuration of wireless communication among mobile devices at home and office environments.

2.6.6 Conceptual Modeling

Conceptual modeling is based on the fact that it is natural to model context derived from the context entities. The assumption for this modeling is that people share understanding of context that revolves around entities. Furthermore, there exist connections or relationships between entities. Entities reveal more relationships or context when they are combined. For example, person, computer, office and time are context entities. With the presence of all four entities, it is conceivable that the person is likely to work on the computer at the office during business hours. The association of these entities reveals the context. When entities are augmented together, the context is clearer. For the same example above, if office entity is not present, then the possibility that the person is working is much lower.

One of the advantages in conceptual modeling is that it is easy to model the entities in a bottom-up approach since it mirrors the way human perceives the world. However, a complete representation of the world by this model is still a blurred image compared to the reality. First of all, the existing sensors and sensing technologies limit the information about entities that the context-aware system can detect and recognize. In addition, the choice or selection of the entities in the model is subjective to the application developer's understanding of context. When entities are aggregated, the number of relationships and contexts which can be derived rises. Therefore, entity aggregation makes the modeling complex.

The conceptual modeling is taken as the modeling approach in this dissertation. Context data is thus assumed to be collected from various sources including wireless sensor network, carrier network, WLAN, and mobile devices. The modeling handles data in different forms, for instance discrete or continuous. It also deals with middle-level context abstraction such as transforming the geographical coordinates into a symbolic address. Unlike other context data models, this context model does not include context inference which derives high-level context abstraction for mobile applications. This is due to the belief that individual applications understand the application-related context better than the general-purpose context server.

This is particularly true when the number of context-aware applications increase and the applications become much more sophisticated. Hence, the high-level context abstraction is distributed to the individual mobile device. The distribution of high-level context abstraction is based on the fact that technological advances make mobile devices seem even more capable. In run time, the mobile application informs the context server of the context data it needs. After receiving formatted context data from the server, the application extracts the high-level abstraction and acts based on the interpreted user context.

The most common data structures used in conceptual modeling are key-value pairs or object-oriented programming. Instead of these ad-hoc data structures, standard XML is used to structure the data in this dissertation. The selection of XML is due to the fact that proposed context server does not have the context reasoning component as other context servers have. Thus, the modeling structure can be much simpler. Context server focuses on the context acquisition and models data into XML format. Since XML is widely accepted in the computing world, XML data can be easily interpreted by different parsers and integrated into different programming languages. The Internet Explorer browser has a built-in XML parser. Most popular programming languages such as Java and C# .net have support to work with XML data structure. A typical example of location in XML modeling is as follows:

```
<location>
        <longitude>x</longitude>
        <latitude>y</latitude>
        <altitude>z</altitude>
        <symbolic>10555 W Flagler Street<symbolic>
</location>
```

2.7 Applications of Context Awareness

This section enumerates some of the creative applications that consider mobile user's context.

2.7.1 Technology Enabling Awareness (TEA)

The European project TEA is a good example of architecture for mobile devices. The objective of TEA is to integrate multi-type sensors on a mobile device (http://www.teco.edu/tea/) [22]. The device architecture separates context acquisition from the individual applications. The diverse range of sensors collects information not limited to location only. The design objective is to have plug-in unit for sensors added onto the terminal architecture. The low level sensor acquires the raw information and passes it on to the context cue. The context cues are the higher layer component in the architecture which is responsible to interpret the raw context. Another goal of TEA is to derive user context from the combined sensor pool. The derived user context should enhance understanding of the user situation. TEA provides a development platform for context-aware applications. It established a direction for self-contained solution in context aware study. After TEA rolled out, a few more projects were built based on TEA architecture. Overall, the TEA project had an impact in the context-awareness research.

2.7.2 Sharing Context Information between Mobile Terminals

The TEA architecture allows context sharing among applications on the mobile device. This application provides a user interface solution that displays the friends' positions in a customized map which runs on mobile devices [23, 24]. The carrier's network functions as the centralized server which updates and distributes individual context information. The server updates group members of other members' activities on mobile terminals. Context aware phone calls and short messages can be sent to the intended persons on the map. The application interface provides easy access for the user to use context-aware applications. This application demonstrates a simple architecture design which enables mobile users in a community to share context information.

2.7.3 Location-enhanced World Wide Web

One of the relevant issues in context-awareness research is the high cost of hardware infrastructure which unduly curbs the hardware deployment in the carrier's network. A good example on how to overcome this drawback is best illustrated by the approach taken by Place Lab. Place Lab built a wireless LAN based on 802.11 location system to assist residents or visitors who navigate in a community [25]. The WiFi positioning algorithm is used to pinpoint the mobile user's position both outdoor and indoor. User's location is integrated into a customized browser. A database stores and updates the location-related information. As the mobile user travels in the community, the user is able to get location-enhanced web service. Because WiFi technology is very cost effective, Place Lab's design is extremely flexible and scalable. The project illustrates the usefulness of context awareness with low cost technology to build the infrastructure. **Figure 2.2** shows the wireless LAN comprises WiFi access points which are in red dots.

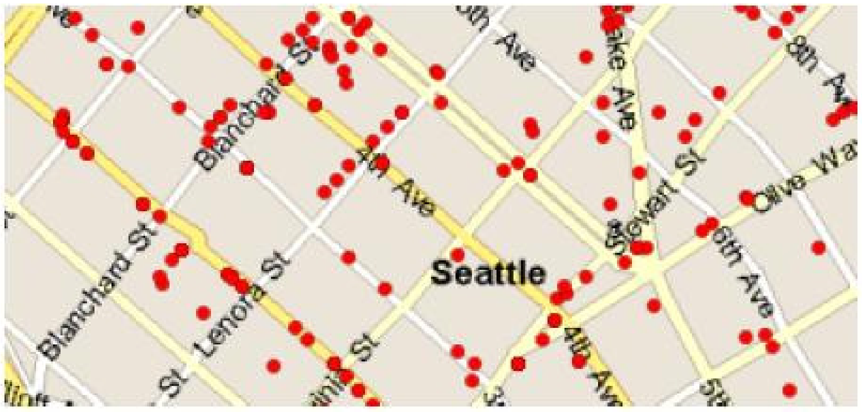


Figure 2.2 WiFi Access Points Distributed in Seattle. Courtesy of UbiComp 2003

2.7.4 Context-Aware Applications in a Hospital Setting

Clinical computer systems, such as electronic patient records, do not use patient's given situation into the system design. Physicians have to manually work with same user-interfaces regardless of the individual patients in different clinical situations. The new designed applications take advantage of physicians' and patients' given situation. The context-aware hospital bed has for example a built-in computer and a mounted display for entertaining a patient or for displaying patient records [26]. This type of context-aware bed is equipped with RFID sensors which monitor activities of the patient on bed and identify the near-by physicians. For example, when the physician comes by for the ward round routine, the context-aware display recognizes the activities and switches the display to patient record from entertainment programming for the patient. This can be combined with the so-called context-aware pill container with a built-in computer which displays the medicine schema to the patient and allows the patient to take the pills from the pill container.

The above list of context-aware applications is not a complete one; it only demonstrates that context-aware research starts to make inroads in different real-world situations with varying computing capabilities. All the applications listed here demonstrate useful and effective context-aware devices in real-world environments.

CHAPTER III

CONTEXT AWARE ARCHITECTURE DESIGN AND REQUIREMENTS

3.1 Introduction

In the world of ubiquitous computing, knowledge and communication are interwoven entities. Pervasive computing refers to the trend toward casual access to numerous invisible computing devices embedded in the environment. Especially the networked computers are so pervasively available to users “anytime and anywhere” without the user interaction. Pervasive computing benefits users in a way similar to electricity. Consumers are all aware of electricity, and yet when they switch on lights, TV, and other powered devices, the act is taken for granted. Likewise, the future context awareness will become the “context aware utility” for mobile users. People use context-aware applications with little notice of the act.

To support such “context aware utility” everywhere, the appropriate infrastructure for context awareness in the environment is essential. Most researches with ubiquitous computing community take the bottom-up approach. The context acquisition at the bottom really affects the accuracy of the context-aware computing at higher levels. Errors or bias in context acquisition will certainly propagate and multiply in the later stage of computing. It seems that understanding of user context is bound to availability and effectiveness of context acquisition. Needless to say, the infrastructure to acquire context is crucial for the success of future context-aware applications. However, the 3G technologies in general have no provision for context-aware issues. The 3G networks can only provide limited context information of the mobile users. As aforementioned, one of the issues in 3G networks is that multiple communication technologies coexist around the world. In North America, WCDMA and CDMA 2000 are going to be the dominating 3G technologies in addition to the existing GSM networks. Meanwhile, WiMax quickly gains the attraction as carriers Sprint and ClearWire launch nationwide WiMax network. With 4G technologies emerging, it is not cost effective to build multiple network interfaces into mobile devices. Clearly, the management and performance of heterogeneous networks become the topic of Next Generation Networks (NGN). Furthermore, context-aware applications running on various mobile devices will increase rapidly in the future.

In the early years of ubiquitous computing, it showed that hardware, especially sensors, became cheaper and smaller, ruled by Moore's Law. One solution for context-aware application lied in integrating sensors on devices for a specific context-aware system. The context-aware application developers designing such single-task system must deal with low-level hardware detail, context acquisition, application functions, and high level user-application interactions. As the context-aware applications increase and become more sophisticated, such an approach soon reached its limit. First of all, the individual application takes different context inputs. FedEx or UPS might be interested in location of their cargo. Other context such as surrounding objects during the delivery route isn't much meaningful to them. A mobile digital TV recorder only needs to know programming schedules and mobile user's topics. That means a lot of sensors must be integrated on-board to accommodate individual applications. In addition, mobile devices with on-board sensors have one common problem. If new sensor or sensing technology emerges, it is not economically or technically feasible to reconfigure the hardware and software on such devices. Furthermore, new context-aware applications might require context information which the on-board sensors cannot supply. Hence, it is not practical to integrate all the sensors. Secondly, the on-board sensors cannot help with context information distribution and reusability. Because developers of specific applications have no knowledge of other applications, as individual applications do not share the acquired context information with each other. Thus, each application requires a set of on-board sensors

The alternative solution lies in context architecture that takes care of acquiring mobile user's context in hardware level and distributing the user context to different context-aware applications via the standardized API. Salber proposed the Context Toolkit for this approach [27]. The main objective of Context Toolkit is to provide object-oriented "widget" which interacts with low-level sensors. One advantage of this widget is that reusability of the low-level context acquisition among multiple applications. Sensor widgets encapsulate low-level sensors and provide context-aware applications access to these sensors. However, the problem with Context Toolkit is that context-aware applications need to generate instances of those widgets on mobile devices. These widget instances consume a lot of system resources such as memory, CPU cycles, and power. With Context Toolkit, mobile applications acquire raw context data, monitor the change in context data, interpreting the context, and adapt to the new user context. The context acquisition is still too tightly integrated with the applications. The acquisition of context takes a lot of system resources. Also, applications are bound to a few sensors at the run time, failure or errors from these sensors might introduce misleading or conflicting user context to the context-aware application. Therefore, the Context Toolkit is not an ideal solution for mobile devices with limited computing resources. Fahy and Clarke proposed middleware, CASS, for mobile context aware applications [28]. This middleware improves over the Context Toolkit because CASS separates the context acquisition from the mobile devices. CASS receives context data from sensors and stores the data into a database. Another feature is that CASS has a context inference component which derives high-level context abstraction for application. Like Context Toolkit, CASS requires applications to generate ChangeListener class and LocationFinder class on the mobile device to detect and track the context change.

Researchers took the context server one step further toward shielding application developers from low level hardware details. Harry Chen soon proposed a context-aware architecture that separates the context acquisition completely from the resource-constrained mobile devices [29]. This architecture which echoes the mainstream design interacts with heterogeneous sensors to acquire user context. Such architecture is also designed to resolve context conflict arising from different sources of sensors. Like other models proposed in this direction, the architecture models the context data with a new technology, and ontology. Ontology is semantic language which is used by computer applications to express and process the description of context. The ontology languages allow information sharing between context-aware agents. A lot of APIs and platforms are built to process such ontology languages. The driving force behind ontology is that context can be expressed or described by such ontology languages. The central component of this architecture is the context broker. The context broker utilizes the web ontology language, OWL, to model and share the context information. Logic inference is used to recognize context and resolve the context inconsistence. The design of the context broker also reflects decoupling of context reasoning from the actual applications. The core of the context broker is based on a number of ontologies. SOUPA (Standard Ontology for Ubiquitous and Pervasive Applications) is used in the context reasoning which is central in the context inference and context modeling. The design objective is to separate the logic inference from the resource limited mobile devices. However, this feature turns out to be the disadvantage of the context server because user contexts are simply too many to cover by the designers of the context server in advance. The other limitation for this broker-centric architecture is designed for the indoor environment, and not for outdoor environment.

Bhattacharya proposed a universal location support system that interprets, represents, and stores location data across heterogeneous networks [30]. This universal location system supports location-aware application features vertical handoff between heterogeneous networks. The Global Location Server and the Network Location Server are combined to map the mobile user’s location given the network id or IP. However, such system limits itself to provide location context only. This architecture has no provision for other user context such as physical conditions in environment and user preference.

Rehman proposed an architecture which is unlike the prevailing mainstream in the context-awareness research field. He designed such architecture based on the model-view-controller paradigm [10]. He believes that humans perceive context as a property of interaction. Objects or people cannot be considered context. Real world objects usually do not have any context until they become part of the interaction between the humans and application.

Figure 3.1 shows the context chain of his architecture. Context generated from interaction enters the chain. Unlike the mainstream context-aware modeling, Rehman's architecture does not have the so called "reasoning module". The context chain needs to define all possible contexts in advance and chain them together. The advantage for the context chain is that the user contexts are further defined and classified with granularity. Therefore, the application behaviors which are associated with such well refined context adapt to user intention much more accurately. However, the shortcoming is that developers must define all possible contexts in order to cover all user interactions. This requirement is certainly a mission impossible from designer's perspective. In addition, the system performance might be an issue. The context flows through the chain until it matches one of the context components in the chain. Then the controller specially designed for that context component is activated. The flow goes on to the models under the controller. As more contexts are added to the context chain, it is clear the context iteration becomes a burden.

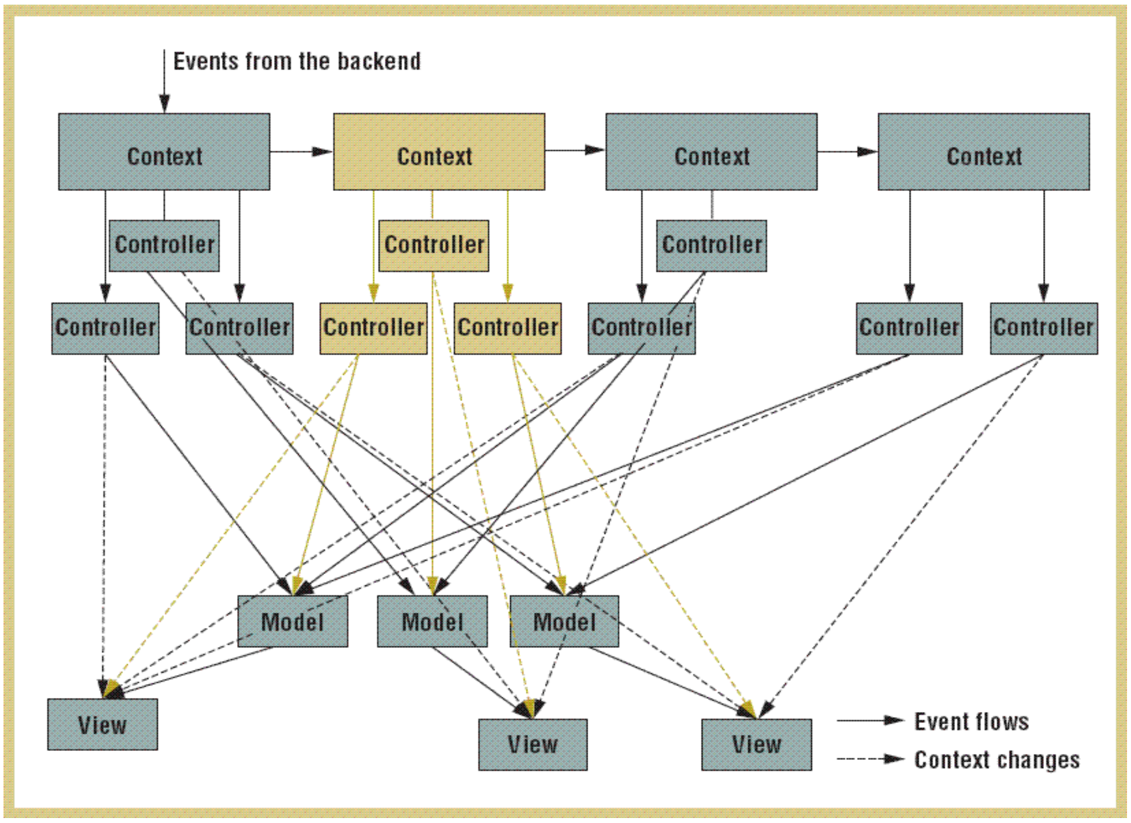


Figure 3.1 Rehman's Architecture. Courtesy of Pervasive Computing

3.2 Required Context Aware Architecture

The context server acquires and processes user context from the low-level sensors or mobile devices. Context-aware applications do not cope with any "sensor widgets" to learn the user context. Instead, applications request context entities through standardized API and data structure, for instance XML. The application developers can truly focus on implementing high-level application without worrying about low level hardware, context information acquisition, and widget's programming structure.

Hence, the required architecture should be flexible to allow dynamic configuration of new context aware applications at run time. Finally, the architecture must be cost effective and scalable over heterogeneous networks. Such architecture requires that wireless carriers or third party must construct the hardware infrastructure and software infrastructure to facilitate context-aware applications. The hardware infrastructure interacts with the hardware devices, collects user context and distributes the user context upon the request of applications. The software infrastructure analyzes the user context, communicates with user applications, and manages user context. The architecture which consists of the hardware and software infrastructures has the following objectives:

- Simplify the hardware logic on the mobile devices
- Provide context information interpretation and abstraction
- Relieve application developers from low-level sensor details
- Manage and share the user context among mobile devices
- Configure dynamically new context-aware applications
- Provide run time context-aware service registry and matching

To meet these objectives, a hardware infrastructure is conceived with appropriate software developments as the different enumerated constraints are kept in mind.

3.3 Hardware Infrastructure

The context acquisition approach developed in this dissertation relies on the data collection that can be provided by any hardware sensors and sensing technology as illustrated in Figure 3.2 Because such hardware architectures are prohibitively costly and difficult to budget, building and deploying such architecture is out of the scope of this dissertation. What is relevant is that when new sensors emerge to acquire user context in a new dimension, new applications that make use of such context information are accordingly developed to provide tailored products and services [31]. Since location information is cheaply available to consumers, location based applications and services have sprung up. For instance, GPS applications are successfully commercialized [32]. Therefore, the context acquisition is the first step to build context aware systems in the bottom-up approach. Understanding of the user context is directly linked to how well systems can be designed to sense and acquire user’s context. The biggest hurdle that keeps wireless carrier or third party from building the hardware infrastructure is the cost constraint. Therefore, a cost effective and scalable hardware infrastructure is key for wide deployment. This research supports a scalable hardware infrastructure design based on the cost-effective IEEE 802.11 WLAN (wireless local area network) as shown in Figure 3.2.

WiFi technology has been widely deployed as the standard Local Area Network at “hot spots” such as office buildings, academic campuses, hotels, and airports. It is feasible for wireless carriers to scale up the infrastructure as the demand rises. Also, it will not be a burden for consumers to bear if mobile devices integrate open technology such as WiFi.

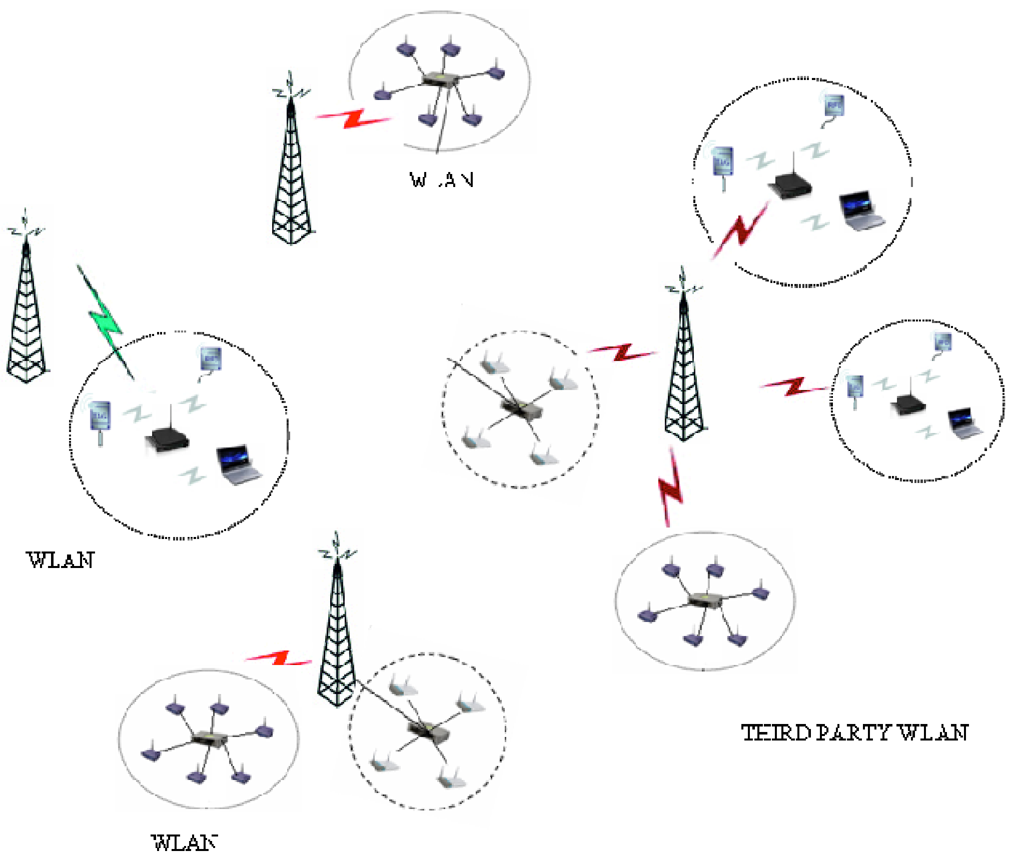


Figure 3.2 Cost-effective and Scalable Infrastructure Based on 802.11

The advantage of this scalable infrastructure is that 802.11 is an open technology which is widely adopted and deployed. The infrastructure can be constructed independently to the underlying 2G/3G networks. The inter-operable heterogeneous architecture works with various communication technologies. Each WLAN is added and connected to the nearby base station as demand rises. However, wireless carriers can alternatively set up the WLAN away from their cellular networks separately. Setting WLAN around base station makes scaling and managing much easier with lower maintenance cost. Various sensors and electronic devices which are connected in such

WLAN can detect and collect mobile user's context information. As mobile users travel from place to place, his/her context information is collected, updated, and disseminated from one WLAN to another.

The manufacturers of mobile devices start to ship devices with built-in WiFi module. Thus, mobile devices which are WiFi-enabled are readily available with reasonable prices. Another advantage for this infrastructure is that mobile users are likely to benefit from the bandwidth which is even better than the latest 3G technologies could offer. For instance, it is cost effective for the mobile users to download hi-definition content from the WLAN rather than the 3G cellular network.

Third parties that are specialized in providing services in certain geographical areas can also register their WLAN with the wireless carrier to offer valued-added service tailored to user's context. Public school, libraries, government branches, banks, and so on could set up their own WLAN and register with operators such as AT&T or Verizon. The wireless carriers regulate and monitor the business activities of the third parties. As mobile users, they can access local resources and experience better services.

In general, mobile users carry out activities at certain indoor areas such as office buildings, stores, schools, and so on. To improve mobile user location granularity and accuracy of other contextual information, further infrastructure is required to set up in places. The cost issue must be considered against the granularity or accuracy of the context information. In addition, the design of the wireless network must cover other requirements for the sensor, such as energy consumption, portability, size, robustness, weight, and reliability.

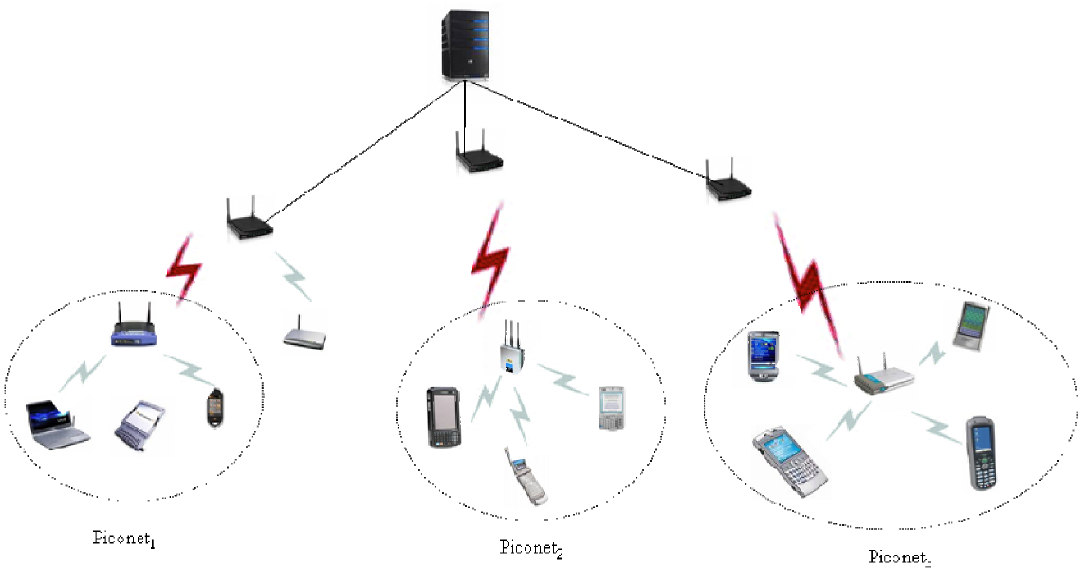


Figure 3.3 Mobile Users Roaming in Piconet

In this research endeavor, a WPAN (wireless personal area network) is assumed as it provides such context granularity. The WPAN is structured as a computer network for communication among the mobile devices which are close to the mobile user [Wikipedia]. In particular, the piconet is distributed within WLAN to collect user context and provide user services as shown in Figure 3.3.

The piconet is based on the open technology, Bluetooth, which features low cost, low power, and small sized radio system. Unlike the infrared beacons, Bluetooth does not require line-of-sight. The Bluetooth technology has been widely integrated or embedded into mobile devices. Clearly, wireless carriers should take advantage of the universal acceptance of Bluetooth. The piconet, as defined in this study, may include sensors, electronic hardware, and roaming mobile devices. Each piconet has one master which is either a sensor or electronic device. In the Piconet, speed sensors, motion detectors, printers, MP3 players, computers are all connected and each can be a potential master of the piconet in order to synchronize the clock cycle. These piconets scatter throughout the indoor environment which is covered by the WLAN. The master provides the identity (hop sequence) and master clock of the piconet. The roaming devices and other Bluetooth-enabled devices join the piconet as slaves by offset individual system clock with the master clock. The master polls slaves and manages the transit in both uplink and downlink.

As mobile user roams out of the range of one piconet, the mobile device is likely to join another piconet. For example, a mobile user downloads ring tone from one piconet. Then this user moves on and locates one Bluetooth printer and prints files stored in his cell phone. Next he travels a few steps and uploads the pictures from a cell phone to one of the Bluetooth image readers which further store the image files into local user account. All the above activities can take place in one building office. The sensors and devices combined can collect user preferences such as type of ring tones downloaded, location granularity, printing jobs, file transfer, and so on.

In addition to the Bluetooth-enabled sensors, there are other types of sensors which can be deployed in the WiFi network provided that they meet the requirements. These sensors can be used to acquire various contexts in the surrounding area such as light, noise, temperature, motion, and movement [33]. For example, a photodiode can be applied to detect light. Digital camera can capture visual images. Passive Infrared sensor can be used to detect heat flow in the nearby area. Microphones can record voice or audio signals in the environment. Technically, all the sensors mentioned here can be integrated with WiFi chip to send signals back to applications running on the server. If these sensors are placed indoor, the energy consumption requirement can be relaxed. One interesting observation is that these sensors can provide more accurate or even additional context data if they are combined in sensor arrays or well-designed group. If a few passive infrared sensors are distributed at right distances and angles, they could well detect moving persons and other objects entering or leaving the covered area. Intuitively, the optimal sensing technology reflects what context is acquired in certain indoor settings. When sensors are deployed in different environment, we must consider the characteristics of each location and select sensors accordingly. The collected user information helps carrier or interested third parties to provide tailored services which are not only limited to location. For instance, similar ring tones can be offered for download from third party for a fee.

In the outdoor environment, the infrastructure is defined by the wireless sensor network (WSN). The wireless sensor network was initially developed for military applications. Wireless sensor network comprises autonomous sensors which are spatially distributed in the environment to monitor physical conditions such as temperature, pressure, sound, vibration, motion and so on. Especially important is the third generation wireless mesh network which overcomes the spectrum contention in 802.11-based systems by using different channels in upstream and downstream radios. Thus, each sensor node is able to send and receive simultaneously to and from its upstream and downstream neighbors [34]. The bandwidth in upstream and downstream is increased. The performance of the third generation wireless sensor mesh is improved 50 to 100 fold compared to the first and second generation. Each individual node contains a radio spectrum robot which monitors other radio traffic and adjusts topology and channel mapping of the network. In addition the 802.11-based wireless sensor network has the following characteristics:

- Small sized, low cost, and heterogeneous sensor nodes
- Node failure tolerance
- Dynamic deployment
- Unattended operation

One of the hurdles in deployment of a large scale heterogeneous sensor networks is that the data packets must hop quite a few nodes before they arrive at the destination. There is a possibility that the data packets might be lost along the multiple hops. In general, the more hops in the route, the worse the performance is. Plus forwarding data packets consumes energy in the sensor nodes. One of the solutions is to overlay another high bandwidth 802.11 mesh network on the wireless sensor network to reduce the packet hops along the way.

When sensor node **A** has data packets to send to destination node **G,** node **A** first searches for the overlaid mesh network. If such overlaid mesh network is present, then node **A** sends data packets to the nearest mesh node of the overlaid network. The overlaid mesh network has much fewer nodes to hop along. Also, the high bandwidth helps to transmit data packets much faster than the wireless sensor network. Clearly, the additional wireless mesh network not only reduces the packet loss, but also conserves the energy in the wireless sensor nodes. Even if the wireless mesh network fails, the wireless sensor network can resolve the data path within the network. Node **A** can simply send data packets to node **B**. Node **B** relays the data packet to its neighbor node **C**. Eventually, the data packets arrive at destination node **G**. Figure 3.4 illustrates the structure of the wireless sensor network.

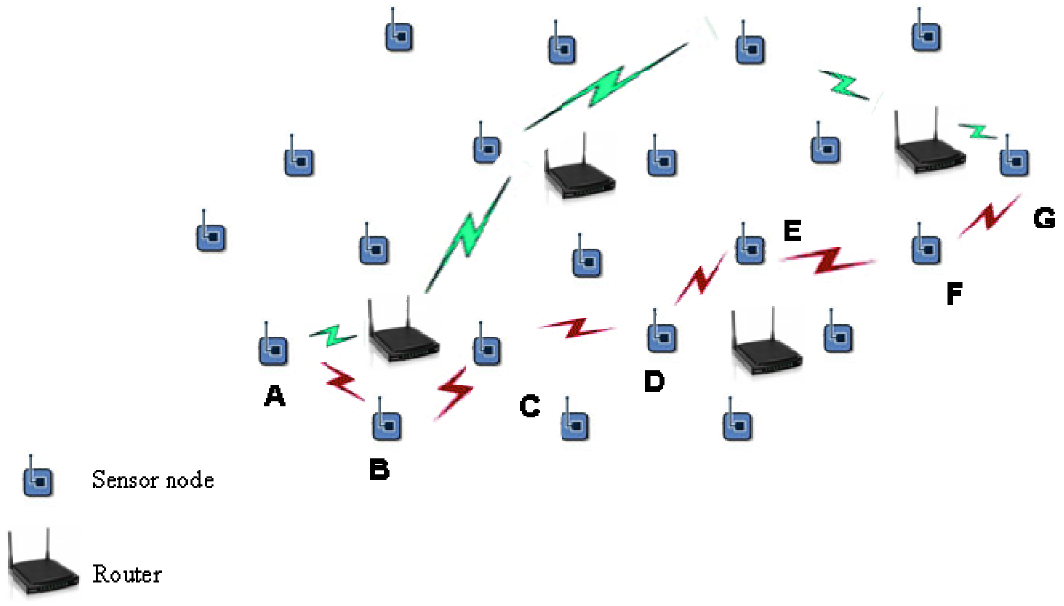


Figure 3.4 The 802.11 Mesh Network Overlaid in WSN

3.4 Software Infrastructure

The software infrastructure is as important as the hardware infrastructure. The required software infrastructure in the architecture given in Figure 3.5 consists of several components, namely:

- The context interpreter,
- service registry,
- service manager,
- query/request interpreter,
- policy registry, and
- context-aware search engine.

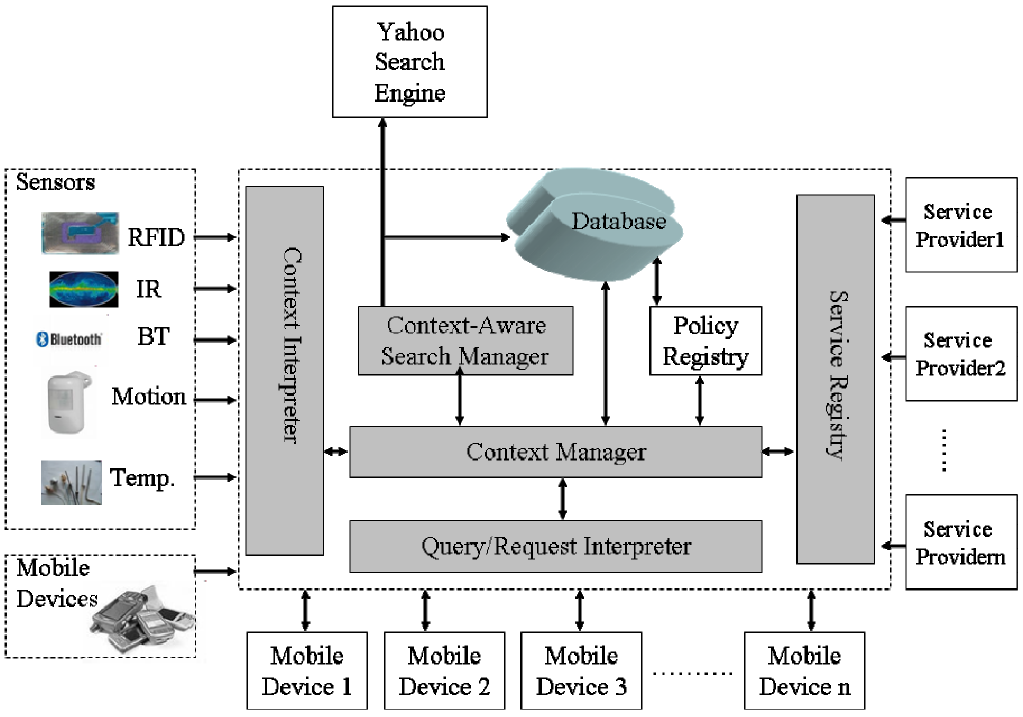


Figure 3.5 Software Architecture and its Context-Aware Server: Blocks shown in gray are modules that were implemented in this dissertation

These relevant components are now described in the different functions they serve and in terms what is expected of them within the framework of this dissertation.

3.4.1 Context Interpreter

The wireless sensor networks provide the context information about mobile users within the covered areas. The context interpreter is responsible for interacting with low-level sensor networks, collecting context information, modeling user context, and updating context entities. This component includes code libraries of procedures for acquiring contextual information from heterogeneous sensor networks. Context interpreter interacts with low-level wireless sensor networks. In addition, context interpreter interacts with mobile devices to receive the user context profiles compiled from the client side.

There are two issues the context interpreter needs to address. First of all, it must deal with the heterogeneity of the ad-hoc wireless sensor network and movement of the mobile user. The sensor network consists of various sensor types, such as RFID (Radio-frequency identification), Bluetooth devices, motion sensors, video cameras, and so on. One of the objectives in sensing technology is to conserve the power consumption of the sensors. For example, wireless sensors can have polling rates quite different to the wired sensors. Sensors in one wireless network do not communicate with sensors in neighboring networks. On top of that, mobile users move from place to place. All these factors might result in conflicting context data from various wireless sensor networks. If a context inconsistency arises from different sources, the interpreter must resolve such inconsistency and update the context profiles accordingly.

The second issue lies in the context data hierarchy. Context data acquired from different sensor networks can be classified into raw context and composite context. The composite context consists of raw contexts. Context interpreter composes composite context based on the requirement or policies provided by the service providers. The acquired context information is further modeled into user context profiles at client side. User context profiles are stored in the database at server side for the retrievals of high-level components in the architecture. The context interpreter also interacts with mobile devices to acquire information as well. Even though the purpose of the required architecture is to shield context sensing from high level context aware applications, mobile devices are a good source for user context. As more advanced applications are made available, multimedia mobile devices offer a user a multitude of applications including email, internet browsing, online game, MP3 players, video camera, calendars, and video streaming. The user data, application settings, and application behavior all reflect user preferences and characteristics. Therefore, sensors are no longer identified as the only source for user context.

3.4.2 Service Registry

The service registry interacts with third-party service providers to register services which will be available to mobile users. Server distributes service providers' information to mobile user according to the user's current context. In a sense, the server acts like a broker who matches services to user's need which is derived from the user context. In this perspective, Figure 3.6 illustrates the service mechanism among server, service providers, and mobile users. Service providers describe services to the service registry with the following format:

- Service type
- Service scope
- Network service and security policy
- Context inputs
- Context change reminder type
- Service description

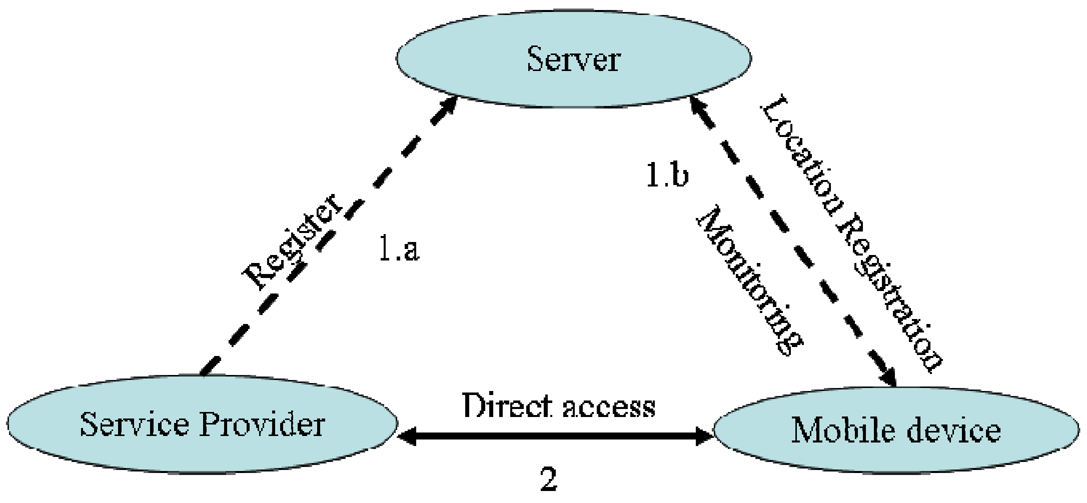


Figure 3.6 Context Service Model

Service type basically describes the nature of the service. For example, a "map" type service provides mobile user a guide for travel. Service type information is used to model the context entities. Usually, the service registry provides guideline for service provider to classify the service.

Service scope specifies either the geographical area in which the service is available, time range, or other constraints. The location context can be specified either in geometric format or in symbolic format. The geometric format models the location in multiple dimensional coordinates. The GPS coordinates belongs to this type of location model. The symbolic format usually models location in a more meaningful fashion. Symbolic format maps local streets, building name, government branches to the geometric coordinates. Besides more meaningful to mobile users, the symbolic format helps store and retrieve location information in context entity object. To demonstrate a service registry, a local car dealer promotes a discount for customers who buy certain models at a specific location for a period of time. This promotion obviously has constraints in time range, location, and price offer.

Network policy specifies the network characteristics of service provider. Service providers have various networks which may have different bandwidth, capacity, and location limitation. Service policy lists all the procedures, conditions, and terms associated with the services. Mobile users with different subscription plans are entitled to different services. Policy can specify login credentials or other required user information and procedures to initiate the service. The security policy on the other hand might list all the user privileges to guard their private information. Users have options to conditionally reveal their personal information to the service providers. For example, a user can choose not to release their personal phone number to unsolicited vendors. Or they can do so in return for a discount on products.

Context inputs list all the required context information from the user. Each context aware application has its own required context. Map providers may be interested in the mobile user’s current location, while other providers might be interested in the mobile user’s gender and age. Location based services are the most available on the market and are generally viewed as most successful. In the future, we will see a lot more services that are not limited to the location. For example, video content provider might well prefer the knowledge of user’s gender, age, and personal hobbies to the current location.

Context change reminder type specifies how the context-aware application is informed of the change in the related context. If pulling mode is specified, then the application itself will pull the context server to detect the change. If event-driven mode is used, then the server will trigger the event to invoke the application. If the user mode is selected, then the mobile user will be prompted for keying the change to inform the application.

Service description basically describes the nature of the service, feature of products, or other information such as warranty. This description will be shown to the mobile users for information browsing. A typical service registry request from service provider is as follows:

```
<Service>
        <type>Online Game</type>
        <scope>
                <location>x</location>
                <location>y</location>
                <constraint>server capacity<constraint>
        </scope>
        <policy>
                <network>WLAN</network>
                <term>subscribers</term>
                <procedure>login</procedure>
                <security>user name</security>
        </policy>
        <context>location</context>
        <context>subscriber</context>
        <reminder>pulling</reminder>
        <description>
        This is the best online game of the year. Only $9.99 per month.
        No time limits.
        </description>
</Service>
```

3.4.3 Query/Service Interpreter

Service interpreter interacts with mobile devices to receive and interpret application requests. It formats the query or request. Next it passes on the formatted requests to the service manager for service matching.

3.4.4 Context Manager

The service manager processes the formatted application requests. If the application requests can be solved with the registered service providers, then the service manager fetches the related information from the database and sends service provider's information to the mobile devices. However, if a user sends search queries that cannot be solved by the service manager, then the user's queries are forwarded to the context-aware search engine.

3.4.5 Policy Registry

The emerging trends of IP convergence and the convergence of heterogeneous networks make the management of Next Generation Networks (NGN) a priority. The context aware architecture must cope with the heterogeneous networks which might be WLAN, GSM, CDMA 2000, WCDMA, or TD-SCDMA. Policy registry stores the management policies the architecture enforces. These policies are important to make services adapt to the underlying network characteristics. In addition, the service provider has policies which are terms, conditions, fee schedules, and user management applied services. Mobile users have their preferences regarding the personal information sharing among the service providers. The policy registry interacts with network management applications, service providers, and user privacy profiles.

3.4.6 Context-Aware Search Manager

As the information exponentially increases on the World Wide Web and beyond, people rely more on the search engines to find the information they seek. The widespread of mobile devices and availability of 3G high bandwidth have certainly expedited the trend for mobile devices to gradually overtake the wired computers in providing users access to digital information. Mobile web refers to user's web access via mobile devices. The mobile web access is hyped as a promising service which will eventually dwarf voice calls and short message services. Wireless users enjoy the access to the mobile web "anywhere and anytime" due to the mobility. Recent research shows that mobile web access provides one additional usage dimension to the mobile users, personal space extension [35]. The personal space extension refers to the mobile web access anywhere and anytime in places with or without stationary PC. Researches in the past saw rising casual browsing during coffee breaks or waiting lines in daily lives. People tend to use mobile devices for private information in office where wired computers are available.

The mobile search started as the wireless extension of the PC-based web search. In the early years, Google built their search engine based on the PageRank method. PageRank method measures global web page rank based on the reference links from other web sites. If web pages hosted on different sites all point to one web site, then this web site is very likely to hold important information for a specific topic. The PageRank method is simple, but worked really well for Google. Over the years, search techniques evolve from simple string match algorithm [36] to sophisticated search on multimedia content such as images and video [37]. The search engine has thus become a critical tool in daily information retrieval. It is estimated that the most popular search engine, Google, currently indexes more than 10 billion web pages. As mobile users need access to information "anytime, anywhere" via smart phones in the near future, mobile search will certainly become the important application on the mobile devices.

Major search engines such as Google, Yahoo, and MSN attempt to increase local content in the query results in the hope that users are provided more information around them. The proposed context-aware search manager improves the mobile search by personalizing the user query. In the traditional query expansion, the user query is expended with additional terms from the thesaurus or a set of documents. The proposed personalized query incorporates the user profiles. The context profiles should help improve the search results, because they provide information related to hardware configuration, user activities, cached data, network traffic, surrounding area, and so on.

The context-aware search manager for mobile devices must, however, cope with some issues not encountered in the wired computers. First of all, hardware is bound to the requirement of mobility. The key pad is tiny and not convenient for typing. For instance, mobile users check emails more often with cell phones than PC, but they delay the email reply until they have the access to the PC. Even though the LCD screen has increased proportionally to the mobile devices, the LCD display size is not going to continually increase beyond the current sizes. The display constraint implicitly requires a search engine to improve the accuracy of returned query results, because query results are limited to fewer references or links than the conventional wired computers. Users are inclined to quit the search, because they have to scroll up and down the pages more than they do with the wired desktop computers. Research conducted by Google has shown that only ten percent of mobile users look beyond the initial set of search results which are typically 10 documents. A field experiment in Japan showed that some mobile users log in Wikipedia to solve their general queries instead of search engine because the Wikipedia has pages formatted to fit for small LCD screen.

Secondly, one of the challenges for the mobile search is to identify the query ambiguity. Short query such as “apple” is not clear for search engine to derive the right topic in user’s point of view. Search engines usually returns documents that cover multiple topics to solve ambiguous queries. Google researchers found that the average mobile query was 2.56 words with 16.8 characters. It takes the mobile users 40 seconds to enter the query words [38]on the mobile device. It appears that users tend to submit short queries. It is likely that mobile users will continue to submit ambiguous queries.

Thirdly, the connection cost is much higher in mobile devices compared to the wired computers. Research also indicates that mobile users tend to use less web applications when they are connected with 3G networks such as WCDMA or CDMA2000 than they do under lower cost WLAN. The cost issue might, in large part, explain why 90% of mobile users do not look beyond the initial search results. This implies that mobile search engine has less user feedback, compared to the PC-based web search.

Fourthly, mobile devices are not only the extension of personal space, but are widely utilized in places where PCs are readily available. There are several studies in the past which focus on the categorizations of the web activities for PC-based and mobile web access. Kellar et al [39] investigated the web activity taxonomies based on the pilot user study, group interview, and field study. They concluded the following major categories: fact finding, information gathering, browsing, and communication. PC-based web search become a necessity for users to accomplish these web tasks. Web search assists user to retrieve information from billions of web sites. Cui et al introduced a new category for mobile user behavior, personal space extension. They studied how people use the web on mobile devices. Based on gathered user data and interviews, they discovered users access mobile web not only for "micro breaks" in places where no PC are available but also in office and home for convenience and faster access. Clearly, mobile devices are used in diverse user context than the wired PC. The mobile search should cope with the user's surrounding area. This would require search functions that include location context and user context in the search results. Google researchers found that more than four percent of all queries are related to the local searches. This will be the trend of the mobile search. For example, the mobile user might find a digital camera at one of the local stores, and a search online can be made to make sure the price tag is competitive.

Mobile search stems from PC-based web search, but differs from PC-based web search due to these aforementioned factors. Mobile search has become the second most used application only after social networking in wireless internet (http://www.opera.com/smw/) [40]. Search engine such as Google appear in top three of most visited web sites in terms of wireless internet usage.

3.4.6.1 Context-aware mobile search

Due to the concept of mobility, mobile devices are utilized in more diverse contexts than PC in office and home. Mobile search is becoming increasingly important for mobile users as mobile devices are more widely used. Mobile search differs from standard PC-based web search in a number of ways: (a) the user interfaces and I/O are limited by screen real estate and key pads are tiny, (b) limited bandwidth and costly connection time, (c) increased local search due to mobility. Most mobile search queries are short due to the hardware limitations. Early studies [41] attempted to provide solutions to mitigate the hardware limitations of the wireless devices. The top 100 mobile queries at AT&T [42] reveal that a great number of search queries are navigational [43] in nature. The navigational searches, for example “Google”, usually steer mobile users to specific web sites conveniently. Unlike navigational queries, informational and transactional words like “images” and “free” are ambiguous to search engine. A housewife and an iPhone user interpret “apple” differently in search context. A housewife is likely to know the apple variety and prices at the local grocery stores. While an iPhone user is interested in service or products related to iPhones.

The user queries are classified into three major categories. The first type refers to queries that cover one unique topic. Navigational queries fall into this category. When people submit “Yahoo”, it is likely they want to access Yahoo’s portal site. Especially, mobile users can save typing time and quickly click on the right URL from search results. The second type of queries is referring to sub concepts or sub topics of one particular thing. For example, when user submits “soccer”, it might refer to a particular soccer game, or soccer boots, or soccer ball, and so on. The third type of queries is ambiguous queries which cover multiple topics. Search engines have hard time solving ambiguous queries. For instance, iPod users want to know the prices of digital mp3 songs from apple web site, while the house wife wants to know the price of Fuji apple at local grocery store.

Therefore, it is imperative to identify the ambiguous query. Researchers studied methods and models to determine the query ambiguity. Clarity score [44] was proposed to evaluate the relative entropy between the query language model and the collection language model.

$$P(w \mid Q) = \sum_{D \in R} P(w \mid D) P(D \mid Q) \tag{3.1}$$

Where *w* is any query term, *D* is a document, *Q* is the query, and *R* is the document set in which each document contains at least one query term. Clarity score (C_S) equation is defined as

$$C_S = \sum_{w \in V} P(w \mid Q) \log_2 \frac{P(w|Q)}{P_{coll}(w)} \tag{3.2}$$

Where V is the entire vocabulary of the collection, and $P_{coll}(w)$ is the relative frequency of the term in the collection.

Large click entropy indicates that user clicks more web pages to solve the query, thus the query is ambiguous. Small click entropy means mobile users have common understanding for a search query. Song [45] developed a classifier to automatically identify three types of queries, ambiguous, broad, or clear query. These methods and algorithms work equally well to identify the query ambiguity in the mobile search.

Researchers have explored personalized search to improve topical relevance of result documents in PC-based web search. Shen et al. [46] studied user's immediate and short-term search context to expand the current query. Qiu and Cho [47] learned user interest from the click history and developed ranking mechanism based on the user interest. Chirita et al. [48] proposed personalized search and summarization algorithms which assist search keywords expansion based on extracted information from local desktop. Dou et al. [49] and Teevan et al. [50] investigated the personalized search strategies and stated that personalization improves the search accuracy on ambiguous queries. So far the personalization is studied only for PC-based web search. Most of such personalization strategies are limited to the user search history, returned search results, and documents stored in PC.

In this dissertation, we extend this line of work into mobile search and derive user context based on profiles which adapt to user location, activities, interaction history, and preferences. Unlike stationary PC, a user can access the mobile web anytime and anywhere. The user context varies, i.e., people, activities, and settings. We further predict that the increased usage of location-based applications and services will lead mobile users to search for local services and information.

Mobile search has come to researchers' attention. Liu and Birnbaum [51] developed LocalSavvy which aggregates local views associated with news events or topics. Vadrevu et al. [52] pointed out the importance of identifying the regional sensitive queries. Hence, the mobile search must cope with local search query.

3.4.6.2 Personalized search query

The context profiles compiled by context proxy at client devices and processed at context server reflect the hardware configuration, user applications, usage history, network traffic, user activities, and derived user situation. Figure 3.7 shows the general design at client device and server. Cache profile, device profile, environment profile, and user profiles are compiled at client device. All context entities in the profiles are updated with context weights. The higher context entity's weight gets, the more consideration is given to that context entity. Of course, not all context entities in the profiles have changing weights. The hardware profile has fixed context weights due to the stable hardware configuration at the client device.

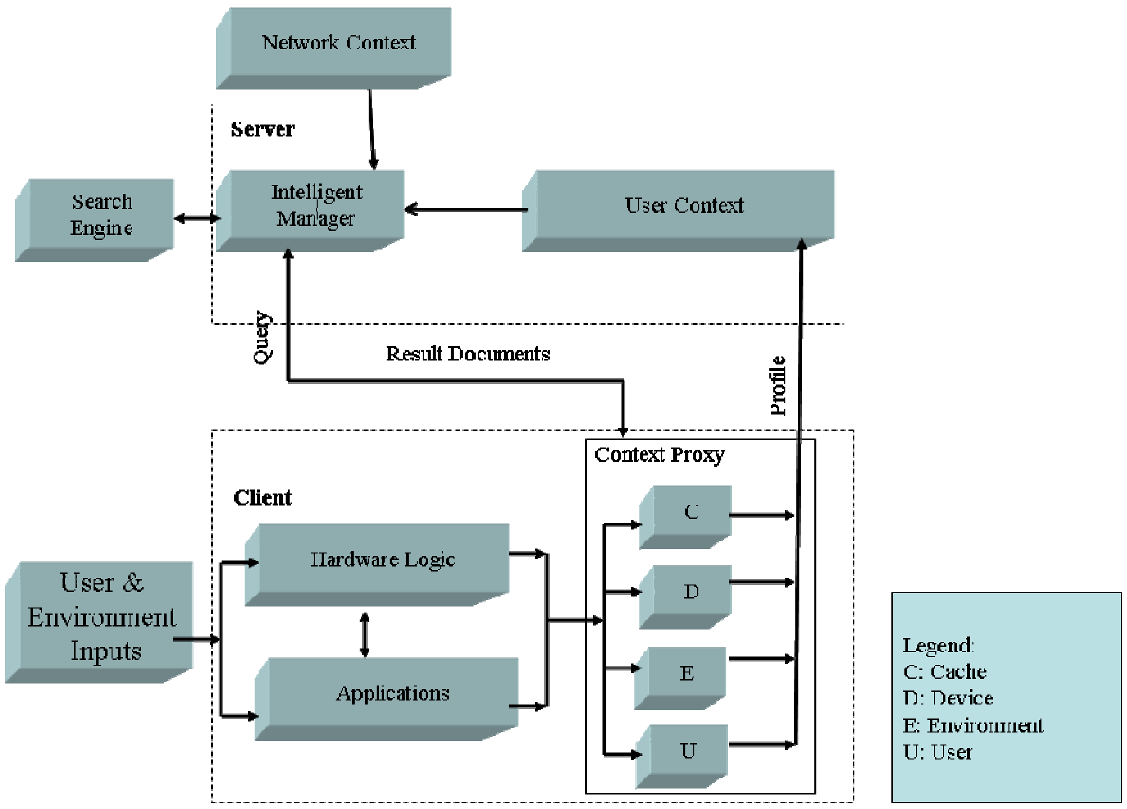


Figure 3.7 System Architecture

Context proxy uploads user profiles to the context server. Intelligent manager at the server monitors mobile user's status by checking the user profiles from client and network. Intelligent manager interacts with search engines via API. Before a query is submitted to the search engine, it is merged with context profiles compiled at server side and carrier's network. The context profiles include the weights of the context entities calculated at proxy, yielding the following form:

$$P = [\mathrm{CW}_i, \mathrm{CW}_{i+1}, \mathrm{CW}_{i+2}, \ldots, \mathrm{CW}_{i+k}] \qquad (3.3)$$

where $i = 0, 1, 2, \ldots, m$. Let the query forms the set, q = [T_1, T_2, T_3... T_n]. After query expansion, the augmented context profile can be formulated as follows:

$$AP = \sum_{i=1}^{m} CW_i + \sum_{j=1}^{n} K(T_j) \tag{3.4}$$

where $K(T_j)$ is the offset function. An example of augmented context profile could be AP = [CW_{i+1}, CW_{i+2}+ $K(T_2)$, CW_{i+3}, ... , CW_{i+n}, $K(T_{i+n+1})$, ...]. If the terms/words from the query set are matched in the context profiles, then the offset weight of the term/word $K(T_j)$ is added to the context weight CW_{i+n}. Furthermore, if the terms/words from the query set are not found in the context profile, then the offset weights of these terms $K(T_{i+n+1})$ are merged to the context profile. It is obvious that the augmented context profile is likely to have more weights on the query words.

3.4.6.3 Reorder Document Set

Google and Yahoo provide API for programmers to integrate search functions into various applications. Especially, Yahoo offers free access to its search engine for programmers. In this dissertation, Yahoo API is thus utilized to solve the mobile user queries. The intelligent manager analyzes user context profiles and sends the modified user queries to Yahoo search engine. Yahoo search engine returns documents to intelligent manager via its API.

Since the returned documents, in general, might cover multiple topics in no particular order, the intelligent manager processes these documents according to the context profile.

It ranks the relevance of the returned documents and reorders the documents in the set. The documents that are most related to the current user context are returned to the user first.

3.4.6.4 User Feedback

Mobile users do not actively provide the feedback to the search engines. Hence, a system is developed to implicitly collect user data. When the user selects the data or services downloaded from the server, his/her selection would greatly help the system understand user's intention. Hence, it is important to utilize the user's selection, as feedback, to adjust context weight calculation at client side. The user selection on the data allows the proxy to reflect the context change into the weights. This feedback mechanism significantly improves the accuracy of the server's response.

Similar to Page-Rank method, context server records all user selection on the search result pages. The selection is a confidence vote from an individual user toward a certain query term. So far human brain is still far superior to most search algorithms in terms of semantic meaning. The pages with most user selection are rated as highly related to a query term. Thus, the page ranking gets higher in the returned document sets. As more users search for the same query, the context server is likely to improve the search result.

3.4.7 Context-Aware Search Application Design Steps

The flow chart provided in Figure 3.8 describes how the context-aware search application solves the mobile user's query.

The steps considered in this context search-aware application include:

Step 1) mobile user submits a query to the wireless carrier's network from the mobile device.

Step 2) Carrier server detects the user query and passes on the query to the context manager.

Step 3) Context manager further sends user query to context-aware search manager.

Step 4) Context manager interacts with context server from the required context-aware infrastructure to requests or updates the mobile user in the mean time.

Step 5) Context server sends user context data back to the context manager. Context manager updates the user context.

Step 6) Context-aware search manager verifies that this is a query to be solved.

Step 7) Context-aware search manager goes on to check if user context data are available.

Step 8) If user context data are available, then Context-aware search manager retrieves and processes the user context.

Step 9) Query manager is activated to augment user search query with current context.

Step 10) Query manager notifies search manager that augmented query is ready.

Step 11) Search manager receives notification.

Step 12) Search manager further submits augmented query to the search engine (i.e., Yahoo) using the Yahoo API.

Step 13) Third party search engine (i.e., Yahoo) returns document set to search manager.

Step 14) Search manager further checks if the mobile user or other users had the same query before.

Step 15) If same query was submitted before, search manager retrieves saved documents related to the query from database.

Step 16) If there is documents set retrieved from database, then merge the documents with search result from the search engine. In addition, documents are reordered. Documents that are relevant to the current user context are listed first.

Step 17) When the document set is ready, notify the search manager.

Step 18) Search manager is notified.

Step 19) Search manager sends the document set to the carrier network.

Step 20) Carrier network is notified.

Step 21) Carrier network passes the document set to the user’s mobile device.

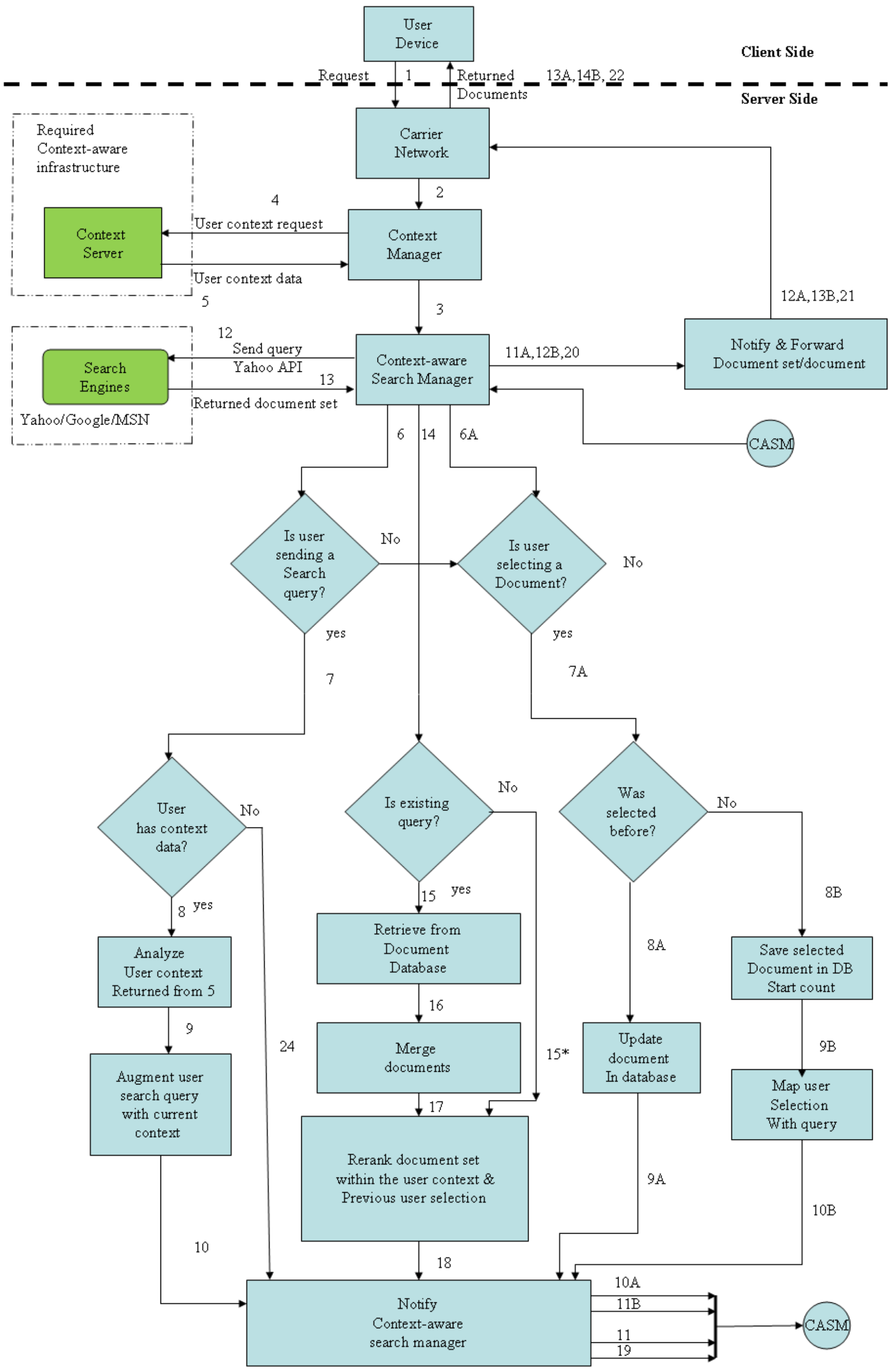

User Device
Client Side
Server Side
Request
1
Returned Documents
13A,14B, 22
Carrier Network
2
Required Context-aware infrastructure
Context Server
4
User context request
User context data
5
Context Manager
3
12A,13B,21
12
Send query Yahoo API
13
Returned document set
Search Engines
Yahoo/Google/MSN
Context-aware Search Manager
11A,12B,20
Notify & Forward Document set/document
CASM
6
14
6A
Is user sending a Search query?
No
yes
7
Is user selecting a Document?
No
yes
7A
User has context data?
No
8
yes
Analyze User context Returned from 5
9
Augment user search query with current context
10
24
Is existing query?
No
15
yes
Retrieve from Document Database
16
Merge documents
17
15*
Rerank document set within the user context & Previous user selection
18
Was selected before?
No
8A
Update document In database
9A
8B
Save selected Document in DB Start count
9B
Map user Selection With query
10B
Notify Context-aware search manager
10A
11B
11
19
CASM

Figure 3.8 Flow Chart for Context-Aware Search Application

The following steps describe how the system responds to the user's selection of the document;

Step 1) User submits a request to view a document.

Step 2) Carrier network detects the request and forwards the document request to the context manager.

Step 3) Context manager redirects the document request to the context-aware search manager.

Step 6A) Context-aware search manager verifies that this is a request for a document.

Step 7A) Search manager further checks if this is the user selected this document before.

Step 8A) If use did select this document before, then update the document count in the database.

Step 9A) Notify the search manager.

Step 10A) Search manager receives the notification.

Step 11A) Search manager fetches the cached document and notify carrier network.

Step 12A) Carrier network receives the notification.

Step 13A) Carrier network sends the requested document to the user mobile device.

If Search manager verifies that a specific document is requested for the first time after step 7A), then

Step 8B) Save the requested document in the database. Initiate the count for the document.

Step 9B) User query and the selected document are mapped to serve future query.

Step 10B) Notify the search manager.

Step 11B) Search manager receives the notification.

Step 12B) Search manager fetches the cached document and notify carrier network.

Step 13B) Carrier network receives the notification.

Step 14B) Carrier network sends the requested document to the user mobile device.

3.4.8 Application Programming Interfaces Based on LDAP

The API is specified to facilitate the communications between the architecture and the service providers. More importantly, the service registry protocols are based on the widely accepted Lightweight Directory Access Protocol (LDAP). LDAP carries out directly over the TCP/IP. This advantage fits well into the IP-convergence trend in the 3G communication. Software developers such as Microsoft and Redhat integrate LDAP into the network technology for directory access. Transport Layer Security (TLS) and Simple Authentication and Security Layer (SASL) which are useful security features provided by LDAP secure data integrity and authentication between service providers and server.

3.5 Context Reasoning

The crucial step after context acquisition is to recognize context from the collected context data. In the early years, researchers build context acquisition, extraction, monitoring, and analysis on the mobile devices. They used pattern recognition to the extracted context against the predefined context. The predefined patterns specify the required context inputs from the environment. The context inputs are further processed to match the patterns. The applications take actions if certain patterns are matched. The problem for this approach is that designers must define the patterns in advance, however the processes for context acquisition, context monitoring, and context recognition are resource consuming at client side. The success is therefore very limited with this approach in a fast changing mobile environment.

Another approach is to centralize the context acquisition and reasoning processes at the context server. Most existing server architectures try to build their context reasoning component which is based on context rules predefined. For instance, Harry Chen makes use of ontology to express or describe such context rules. The context rules are usually deposited into a persistent knowledge base. The context server determines the type of context after data acquisition and decides if context data can be interpreted or analyzed by the rules in the knowledge base. The context rules are loaded to deduce high level context abstract from low level data. For example, compile raw context data into composite context and recognize the context to derive user intention. The extracted user context is searched along the context chain until it finds a match which triggers certain mode. Obviously, the advantage for such reasoning component is that it simplifies the application design on resource-constrained mobile devices.

The dilemma for such an approach is that if the context rules are very specific, then such rules are very helpful for certain applications. However, the detailed rules lose general coverage for other applications. If context rules are general and less specific, they cannot help to narrow down the user context for individual applications. Most researchers choose to specify rules in details to demonstrate usefulness for individual projects. If all the detailed rules are loaded into the knowledge base, it becomes a challenge for the context server to extract context pattern and load proper context rules. Even though the context rules specify the context abstraction and various technologies such as ontology are brought in to help the description, it is possible that a lot of existing context rules only vary a little. Imagine the context server has to search along the context chain in Rehman's architecture. Clearly, Rehman's architecture does not scale well if the context chain continues to grow. Likewise, Chen's architecture faces the same issue. So the performance of context server with built-in context reasoning component is in question if such context server has hundreds of thousands of context rules and interacts with millions of user applications simultaneously.

The approach proposed in this dissertation tries to strike a balance between the context server and mobile devices. The context acquisition is centralized at the server to ensure the reusability of context information among mobile devices. However, the context reasoning remains at the application level. This approach is based on the following assumptions. First of all, the application understands context in question better than a general context server would. This is so because individual applications are designed to act on a small set of context data and target a specific task, while the context server has no idea to address a specific issue at the client side. Secondly, context reasoning takes less system resources than context acquisition. Whenever the context-aware application needs the latest context, it sends a request to the server and receives an updated user context. Hence, system resource, for example power and CPU cycles, is saved at client side.

Thirdly, the advance in hardware makes mobile device more capable to handle increasingly sophisticated applications. The Moore's Law applies to mobile devices. In the foreseeable future, it is anticipated that the hardware capability on mobile device will match the one on computers. The context server does not make a decision for individual applications on mobile device which a context user is in. Instead, the mobile device deduces the user context and updates the context server of the user context. Hence, a centralized context acquisition and distributed context reasoning should be a better solution overall.

CHAPTER IV

CONTEXT SOLUTION AT CLIENT SIDE

4.1 Introduction

Online video streaming, mobile TV, internet browser, emails, calendar, and short messages are all typical applications that become part of a mobile user's life. As mobile users interact with these applications, they collect user information and store user settings which can be used to analyze mobile user's context. The pattern users make on the basis of certain applications can also provide information to analyze a user.

The progress in IC design makes sensors smaller. The on-board sensors are used mainly to improve the human-machine interfaces. Schilit designed a cell phone which integrated tilt, light, head, and other sensors. These sensors are used together with the application software to derive whether the cell phone is on the table, in the suit case, or in an outdoor setting. For example, the management software can automatically adjust the ring tone and volume according to the position. The iPhone has touch screen which replaces the key pad. It also has a header sensor which detects how close the user head gets to the phone. Light is switched off to conserve the power when head is close to the phone. Some smart phones require user's finger print to unlock the screen. Cell phone manufacturers will continue to integrate different sensors to enrich user experiences.

As the mobile devices become more sophisticated, mobile devices are a good source to collect user's data, preferences, characteristics, activities, and profiles. Unlike most existing context architectures, the required approach tries to strike a balance of responsibilities between the context server and mobile devices. The context server undertakes the context data collection from various sources including the mobile devices. Thus, context server shares the user context among networks, user devices, and applications. The context server can provide mobile devices context data the devices cannot acquire from the on-board sensors and applications. In addition, the mobile applications become increasingly important source for user data. The ability of acquiring user data at client should not be taken lightly. The mobile devices should play a role in collecting user context data given the increasingly capable applications.

Because of the variable and instable nature of user context, the difficulty in context awareness at client side lies in the extraction of useful feature/context from changeable user situations. It is therefore important to design new algorithms at the client side to undertake the preliminary context analysis of user situation. The algorithms or client proxy should relieve the networks (i.e., servers) of the computing burden and reduce the need for uplink bandwidth [53]. The purposes of the user context profiles managed by the client proxy are as follows:

- Filter out irrelevant user information and describe the user's current context.
- Update server of current user context.
- Assist in predicting user intention at both server and client side.

Upon receiving the context profiles from the client devices, the server could provide context-aware content/services based on the context profiles.

In the past, Artificial Neural Networks (ANN)-based algorithms have elicited models in context-aware mobile computing applications [54]. A few models have been proposed as exemplified by Clarkson and Pentland who carried out experiments with unsupervised clustering method, Hidden Markov Models (HMMs), to verify the audio and video contextual events [55]. Researchers at Nokia research center utilized a lattice-based structure, Symbol Clustering Map (SCM), to update the weights which are provided for context symbols. The updated weights quantify the context symbols so as to recognize the particular context [56]. In addition, other methods such as Self-Organizing Map (SOM) [57] and Markov Models (MM) [58] are applied or combined to overcome the mentioned difficulties in assessing context.

The required client-side architecture exploits dynamic neural network based on a learning of a single layer net. Weights in context profile at current epoch can be updated with respect to the previous epoch. The merit of a learning rule is that it is simple and easy to implement. As portable devices become more capable of delivering multimedia content, mobile users are likely to increase their usages of these devices for digital data such as Internet access, video, and mobile commerce (m-commerce), to name a few. The surrounding environment may change from time to time due to the user's mobility. One of the challenges in the information retrieval for mobile users is in establishing a context sensitive retrieval process. Furthermore, the mobile search for information on portable devices should match the web searches. Currently, most web search algorithms are limited solutions for context sensitive retrieval and mobile search. This is so because most existing algorithms do not take into account mobile user context inputs such as surrounding environment. For this reason, a context-aware search scheme at server side (i.e., carrier's network) is devised to provide content based on the compiled context profiles.

4.2 Required Client Side Architecture

The traditional client/server paradigm fits well into the context-awareness applications. The carrier's network functions as the server that provides data and voice services to subscribers. Mobile devices request services from the network. What makes this approach different to other models is that the mobile devices play an extensive role of collecting, analyzing, and extracting context entities. Context profiles compiled at client side greatly reduce computing burden at network/server side.

Figure 4.1 demonstrates the structure of the paradigm for the client/server model. The user inputs (voice or digital data) and surrounding environment inputs (temperature, position, altitude, etc.,) are collected by the hardware logic or the applications such as the operating system. The context-aware proxy further inspects inputs and extracts context entities from the inputs. Finally, the proxy compiles the context profile and sends it to the network/server. The network/server learns the user situation using the context profile in addition to the data collected at the network. Server application provides services and data to client based on the learning of the user situation.

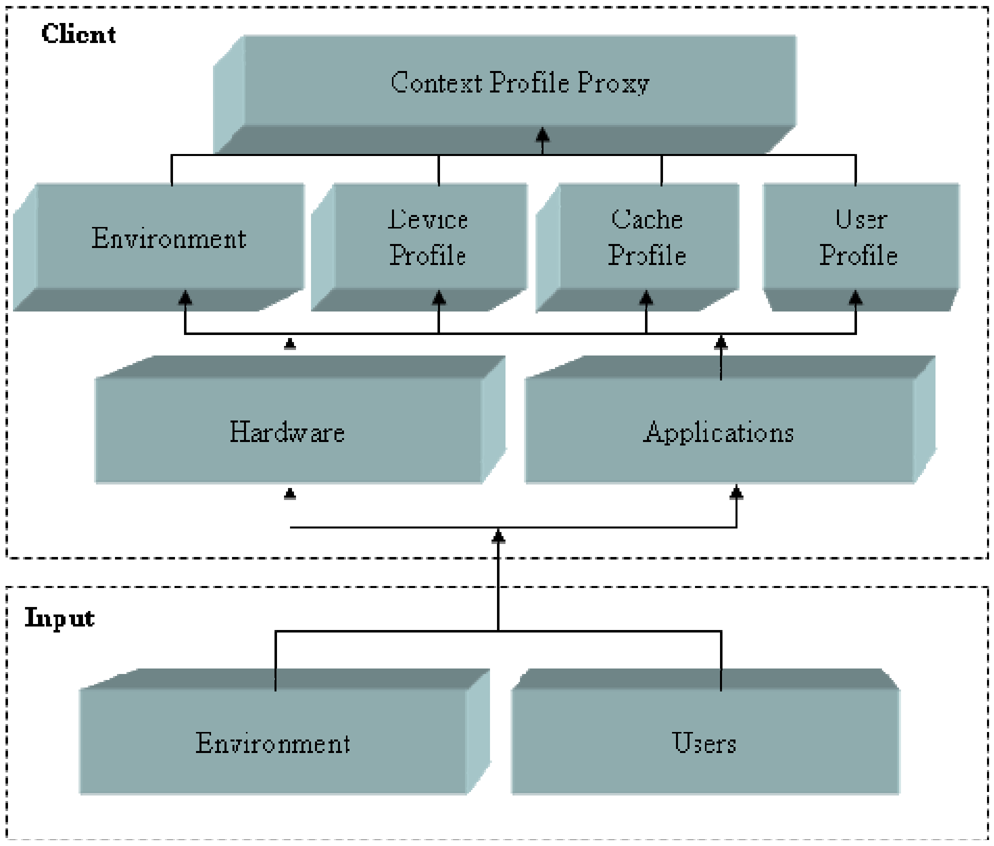


Figure 4.1 Client-Side Architecture

4.3 Client Side Requirements for Context-Aware Solutions

The client proxy runs in the user devices monitors and collects mobile user information through sensors and applications. In addition, the proxy further compiles the context profiles which adapt to and reflect mobile user's changing situation. Cache schemes, virtual frame, and reference frame are thus proposed to improve the efficiency of the proxy's operation at the client side.

The context profile is a collection of context entities extracted from the on-board sensors, client applications, user activities, and so on. Ideally, context profiles should be a good reference of subscriber's current situation from which the carrier's network could derive subscribers' context or intention. Thus, the proxy on mobile devices frequently updates context profiles and uploads them to network for reference if necessary. Due to the limited uplink bandwidth [39], the context profiles should be concise. As user changes activities/context, context entities are added to or dropped from the profiles based on the algorithms which will be elaborated upon in section 5.6. Whenever an addition or deletion of context entities occurs, the proxy notifies the network of the changes. If the user's situation remains the same or change a little, then only updated weights in the context profiles are sent to the network. There are four context profiles managed by the client proxy:

- User profile
- Device profile
- Environment profile
- Data profile

Device profile describes the configuration and features of the hardware. Because the device hardware configuration is quite fixed with respect to user activities, the weights for entities in device profiles are relatively lower and remain unchanged. They are included mainly for server to deliver content in a way that fits best for the device.

4.4 Cache Scheme for Client Proxy

Least Recently Used (LRU) replacement algorithm [59], due to its simplicity, is widely adopted to manage the data cache in the traditional computing such as database system, file systems, or CPU design. The frequency-based replacement algorithm [60] factors out the locality from the reference count. Reference count remains 0 until the data block ages out of the "new" sector. The frequency-based replacement algorithm is effective to filter out bursts of reference counts. These data cache schemes, somehow, are not fit for wireless systems. Furthermore, these algorithms do not take mobile user's context into account. This dissertation proposes instead a new data cache scheme based on the user context profiles, virtual frame, and reference frame simultaneously.

When cache misses happen, the mobile devices do not have the information mobile users require. The downloaded data from the server must be cached in the local memory. With the traditional data cache scheme, the data is stored in the unit of memory block. In our cache scheme, the client proxy examines the data and extracts the incoming data into individual context entity based on context profiles defined early. The recognized context information is stored specifically in a context cache. Each cached context entity is coupled with a weight pointer. The weight pointer points to their historical values recorded in the virtual frames. Figure 4.2 shows the structure of the context cache. Each context entity, C_i, has its own weight pointer, W_i, which points to the history value in virtual frames recorded at different times.

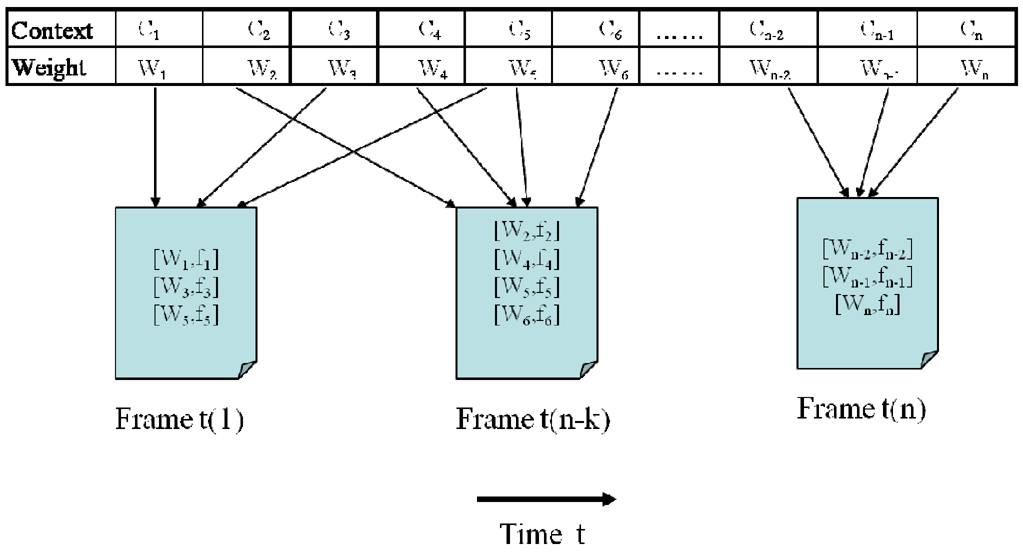


Figure 4.2 Structure of the Virtual frame

In this dissertation the concept of virtual frame is introduced. The idea of virtual frame comes from the movie industry. Movie films consist of individual frames that are played at a constant rate, for instance 24 frames per seconds. Each frame captures the physical scene of the context that leads to the understanding of the situation at the particular moment. Likewise, the virtual frame expresses the meanings of the mobile user's context by featuring weights that are most significant at a particular moment. In other words, the virtual frame does not store any graphical components but weights of the cached data that captures the context of the mobile user at specific time frames. Figure 4.2 illustrates this concept. For example, virtual frame f_1 consists of weights that are recorded to capture the context of the mobile user at time t_1. Likewise, virtual frame f_n contains all meaningful weights that describe the mobile user characteristics and environment at time t_n. Time plays a great role in predicting the mobile user's context or intention. Intuitively, the user's context is likely to remain unchanged or remain consistent over a short time interval in most scenarios.

With the virtual frame defined, the notion of reference frame can be introduced. The reference frame only packages data cache weights as virtual frames. However, a reference frame is not like a virtual frame in that the reference frame is a moving average of the virtual frames. The reference frame averages the most-recent N virtual frames, as shown in Figure 4.3, depending on the configuration of individual mobile devices.

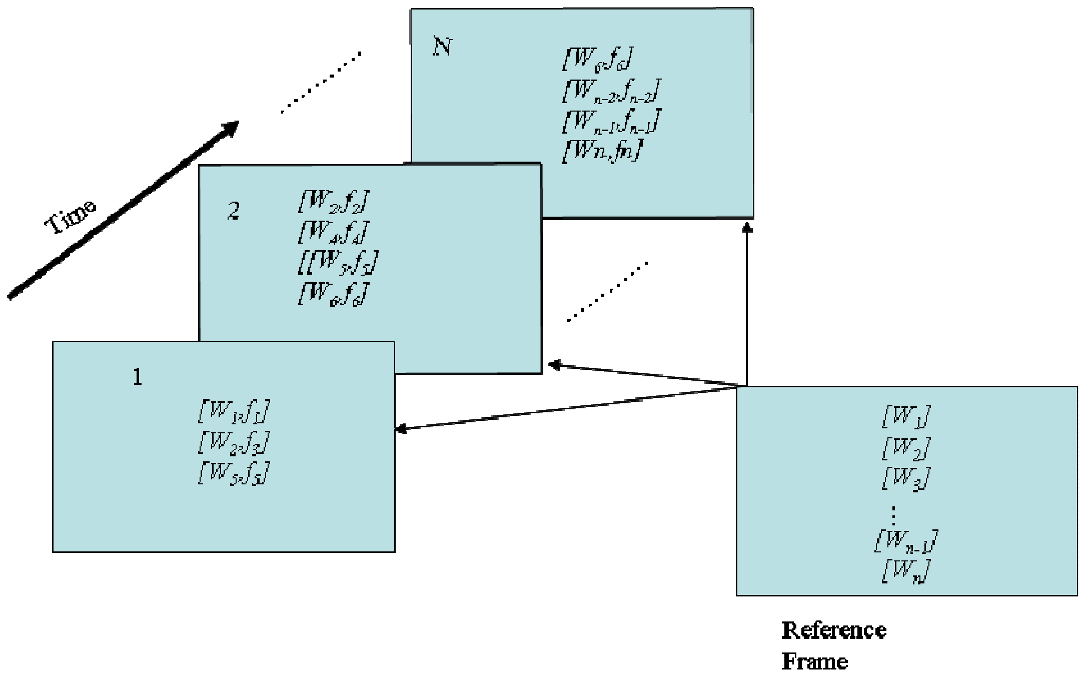


Figure 4.3 Structure of the reference frame

Weights included in the reference frame are calculated as shown in equation 4.1:

$$W_k = \tfrac{1}{N}\sum_{f=1}^{N} W_{fk} \qquad (4.1)$$

Where f indicates the virtual frame number as $f = 1, 2, 3, 4, \ldots, N$; while k refers to all individual weights contained in N virtual frames as $k = 1, 2, 3, 4, \ldots$

It is likely that a specific weight, W_k, might not be captured in every virtual frame, f_i, as mobile user's context changes over the time. Clearly data cache weights that appear most frequently in virtual frames would be the dominating weights appearing in the reference frame. Thus, the reference frame is a much more balanced frame that describes the current mobile user's context with respect to both present and past. Definitely, there would be a tradeoff between the history context and the current context. As N increases, more virtual frames in the past would be included for consideration. So the latest virtual frame has less bearing on the reference frame. Therefore, the reference frame derived from a large set of virtual frames would account for more consistent context for mobile users. On the contrary, if N decreases, then the most recent virtual frame weighs more on the reference frame which better fits mobile users changing their context dramatically in a short period of time.

Obviously, it is desirable to make N adaptive to the context change. If the client proxy figures out that the mobile user's context remains relatively consistent meaning no new context entities are brought into the most recent virtual frame, then N is incremented up to the total virtual frames in memory as in relation 4.2:

$$N = N_{previous} + 1 \tag{4.2}$$

where $N_{previous}$ refers to the total number of virtual frames used to derive the reference frame in the last iteration. By increasing *N*, the reference frame would take more history virtual frames into consideration if context changes were little.

If the client proxy detects that the most recent virtual frame will bring in new weights that do not exist in the current reference frame, then *N* changes according to equation 4.3:

$$N = \left\lceil N_{previous} [1 - \frac{k(w_{new})}{k(w_{reference}) + k(w_{new})}] \right\rceil_{ceiling} \tag{4.3}$$

where $k(w_{new})$ counts new weights that do not exist in the last reference frame; $k(w_{reference})$ sums the number of total weights recorded in the last reference frame. Next, the proxy rounds *N* by taking the ceiling value. As *N* decreases, the last virtual frame could weigh more in the new reference frame. Thus, the cache system adapts much faster to the mobile user's context which changes dramatically over a short time.

When a mobile user queries information that is already stored in the local memory, it is called a cache hit. The client proxy further inspects to see if this is a context-awareness hit, meaning that one or more context entities stored in the context cache is (are) required by the user again. In a context-awareness hit, the client proxy would increment the reference count accordingly, generate a new virtual frame and update the reference frame. Similarly, if the client proxy finds that it is a context-awareness miss, then new context entities and their respective weights would be stored in the context cache. Sequentially, these new weights would be reflected in a virtual frame and the updated reference frame.

If the client proxy detects that the local memory is full, then cache pruning must be carried out. With frequency-based replacement, the cache block with the least reference count could be chosen for deletion. LRU simply replaces the least recently used memory blocks. In this work, the proposed method picks the virtual frame as the candidate for cache replacement. The client proxy calculates the context distance for each virtual frame. The context distance, d_i, for virtual frame, f_i, is calculated as:

$$d_i = \sum W_{reference,p} + \sum \left| W_{reference,k} - W_{i,k} \right| + \sum W_{i,q} \qquad (4.4)$$

where $\sum W_{reference,p}$ sums all the weights that exist in reference frame but not in the virtual frame f_i; $\sum \left| W_{reference,k} - W_{i,k} \right|$ totals the difference of weights contained in both virtual frame and reference frame; $\sum W_{i,q}$ calculates the total weights that exist in the virtual frame, but not in the reference frame.

Context distance should indicate how far away a particular virtual frame is to the current reference frame. If d_i increases, then the context reflected by virtual frame f_i at time i has shifted away from the user's current context. On the contrary, a smaller value in d_i signals that virtual frame f_i is still a close assessment of the current user context. The virtual frame that has the largest context distance would thus be the first candidate for replacement. Next the weights stored in the candidate frame but not in the reference frame should be deleted from the context cache, because weights in the reference frame are within the current user context. After deletion of the virtual frame with the largest context distance, if the mobile device is still low in memory, then same logic would apply to the virtual frame with the second largest context distance.

4.5 Implementation of an Artificial Neural Network for Deriving the Weights of the Context Entities

An artificial neural network (ANN) was implemented to derive the weights for the context entities described in Chapter II. For the implemented ANN, the learning rule is based on the following general equation:

$$\Delta W_{ij}(t) = x_i(t).y_{i(}t) \tag{4.5}$$

In this case, $x_i(t)$ is defined as a function of the context changes ΔC and frequency count ΔF of the context entities in the context cache managed by the client proxy between the current epoch and the previous epoch. In addition, $y_j(t)$ is defined as the weight values, $W_{refernce}$, in the current reference frame.

Thus, the derived equation of weight change for our approach at epoch t is given by equation 4.6:

$$\Delta \mathrm{W_k}(\mathrm{t}) = \left|\Delta \mathrm{C(t)} + \mu \Delta \mathrm{F(t)}\right|_k W_{k,reference}(t) \tag{4.6}$$

where $\Delta C(t)$ is calculated as the ratio given by:

$$\Delta C(t) = \frac{|C(t) - C(t-1)|}{C(t-1)} \tag{4.7}$$

$C(t)$ in this case is the current context value, and $C(t-1)$ is the previous context value. If $\Delta C(t)$ is zero, then this same entity is recomputed as the following absolute difference:

$$\Delta C(t) = -\left|C(t) - C_{average}\right| \tag{4.8}$$

The $C_{average}$ value in this case represents the mean of the elements $C(1)$ through $C(t-1)$. The term $\Delta F(t)$ is the difference in frequency counter for context entity between the virtual frame at current epoch and the virtual frame from previous epoch, $\Delta F(t) = F(t)$ - $F(t-1)$. μ is the coefficient that is machine specific.

Finally, the weights for context entity k at epoch t are updated as

$$Wk(t) = W_k(t-1) + \Delta W_k(t) \tag{4.9}$$

If the virtual frame brings in new context weights that do not exist in the current reference frame, then the initial value of $W_k(t)$ is set to 1.

4.6 Design Steps for the Proxy at the Client Side Including the Cache Scheme

The flow chart given in Figure 4.4 describes how the client proxy works with cache scheme.

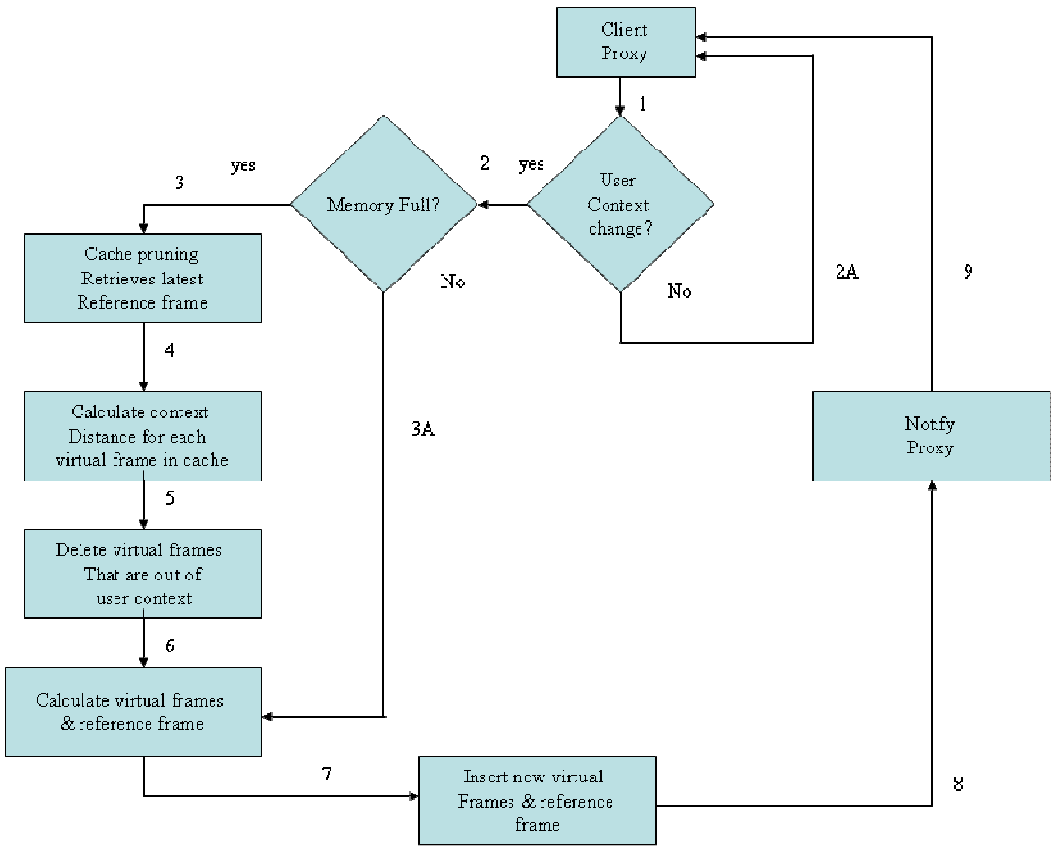

Client Proxy
1
User Context change?
2
yes
Memory Full?
yes
3
Cache pruning Retrieves latest Reference frame
4
Calculate context Distance for each virtual frame in cache
5
Delete virtual frames That are out of user context
6
Calculate virtual frames & reference frame
7
Insert new virtual Frames & reference frame
8
Notify Proxy
9
No
No
2A
3A

Figure 4.4 Flow Chart for Proxy with Cache Scheme at Client Side

From this flowchart, the following steps are undertaken to show how context change is monitored and how virtual and reference frames are calculated, and when necessary how the cache is pruned.

Step 1) Client proxy detects if the user context has changed.

Step 2) If the user context has changed, then check if memory is full.

Step 3) If memory is full, start cache prune. Retrieve the latest reference frame.

Step 4) Calculates the context distance for each virtual frame in the cache.

Step 5) Delete virtual frames based on the context distance calculated. The higher the context distance is, the more likely that the virtual frame is out of the current user context.

Step 6) Calculate virtual and reference frame.

Step 7) Insert new virtual frames into the memory.

Step 8) Notify the proxy.

Step 9) Client proxy receives the notification.

If there is not user context change at step 1), then no action is needed.

Step 2A) Client proxy stands by.

If memory is available at step 3), then jump is made to step 6).

CHAPTER V

EVALUATION OF THE CONTEXT-AWARE DESIGN

5.1 Software Platform

Sun Microsystems created the slogan "Write once, run anywhere" to highlight the merits of Java at crossing different application platforms. Over the years, Java gained popularity within the programmer community. Java applications have made inroads into online applications, wireless devices, and consumer electronics. Java has gained Sun Microsystems more fame than their UNIX servers so that Sun Microsystems changed the stock tick to Java.

Java platform is consequently selected in this dissertation to simulate the test cases on both client and server sides. Java 2 Micro Edition (J2ME) is a programming language for writing the applications on the mobile devices. Eclipse is an open source Java programming IDE suited for J2ME coding. It supports the latest version of Java Wireless Toolkit. Eclipse SDK 3.4.1 is used to editor the J2ME applications. Sun Java Wireless Toolkit 2.5.2 is used to run the J2ME applications.

J2SE 6.0 which features Netbeans IDE 6.10 is selected to write Java applications for the server side. Java programs are compiled into bytecode which can then run on Java virtual machine. Java virtual machine can be installed on Windows, UNIX, and Linux operating systems. Netbeans IDE is one of the free distributions for Java programmers.

These combined advantages make Java platform most suitable for this dissertation project.

The database software is MySQL. The version is MySQL 5.1.30 which includes features like stored procedure, trigger, and query caching. A Swedish company, MySQL AB, started MySQL in the 90's and later acquired by Sun. Since then the MySQL has become even more popular. According to MySQL AB's website there are more than 11 million installations around the world and 50,000 downloads everyday. It is estimated that MySQL has gained 25% market share of the database. Like Java, MySQL can run on multiple operating systems such as Windows and Linux. MySQL server is available to individual developers as free software under GNU General Public Licenses. The MySQL contributes to the open source which changes the landscape of the software industry. The software developed with Java and MySQL combination should make the applications portable, which constitutes another intrinsic capability of this project.

Yahoo Search API is utilized to integrate the search function into the simulation. At this juncture, the Yahoo Search API is gratefully still open to the public access free of charge. Yahoo search site takes the query as the input and returns the documents in XML. XML parser is required to interpret the search results. Intelligent manager ranks these documents and reorders them based on how relevant they are within the current user context. The Yahoo search engine URL is:
http://api.search.yahoo.com/WebSearchService/V1/webSearch?appid=YahooDemo

5.2 Evaluation Strategy

The ideal prototype would have required significant resources in order to build the wireless sensor networks, Piconet, WLAN for services, and carrier's network in order to independently evaluate the merits of the proposed design strategy. Obviously, this is not possible for an individual dissertation to set up all the required testing environment without the impositions of a heavy cost for establishing such a comprehensive infrastructure. Therefore, the software simulation using a free Yahoo API service is thus provided as means to evaluate the effectiveness of the design approach in a most realistic way, alleviating at the same time the prohibitive cost and time demands of these hardware requirements and their eventual deployment.

The context data acquisition is not included in the evaluation, because context data acquisition requires the hardware infrastructure. Rather, all the needed context data are preset and presented in a machine readable format. It is assumed that the wireless sensor networks will function as intended. In addition, data modeling and data conversion do work as proposed. The focus is placed on the potential day to day scenarios. This same approach is considered by many researchers investigating context awareness prior to deployment as a cost effective measure to assess the merits of a potential architecture design.

Vendor GUI is provided for vendor to register services. Vendor GUI collects service information and sends it to the context server. The services will be later provided to mobile users based on the user's context profiles. Client proxy is programmed to analyze the user context data collected on the mobile devices. An ANN-based algorithm is designed to calculate the context weights. Context profiles are compiled to reflect user's current situation. Context search engine is simulated to take advantage of user's context profiles. The context search engine combines searching algorithms with context profiles.

Figure 5.1 shows the simulation flow of information tying all the essential elements of the system architecture design. In general, a mobile user sends a query from a mobile device to the server. Server works on the user query and forwards it to the Yahoo search engine. Yahoo search engine returns the document sets back to the server. The server processes the document sets and sends them back to the mobile device.

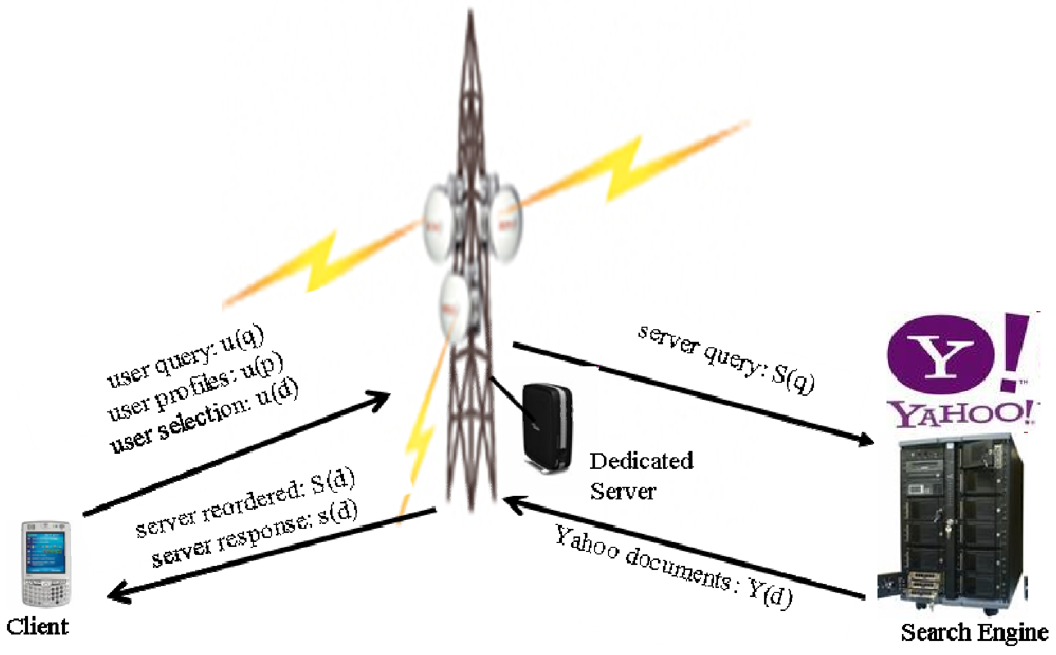


Figure 5.1 Simulation Flow of Information and Elements of the Architecture Design

Proxy is coded at the client side to calculate the context profiles. Also GUI is implemented at the client device to handle the user queries and document selection entered from keypad. An XML parser is added on the device to parse the XML data, mainly document(s), from server. On server side, the search manager is coded to handle user queries. In addition, the search manager modifies the user queries based on the context profiles. The XML parser and Yahoo search API are utilized to handle returned documents from the Yahoo search engine. Search manager reorders the returned documents in relevance to context profiles. Server forwards the reordered documents to the client device. Server also processes user selection on the search documents. User feedback mechanism updates user selection which will affect future document ranking. The feedback mechanism improves to solve ambiguous queries.

User Profiles u(p): The client device will update the server from time to time. The context profiles are uploaded to the server as the user context changes. The profiles derived from the reference frames are stored in the database on the server. Context manager at the server side could forward the user context profiles upon the request from context-aware applications. MySQL server is utilized to store the context profiles

User query u (q): When user submits the search query to the context server, the user query is sent to server in XML. An example would be:
<request>search</request><value>phrase</value>.

User selection u(d): Server forwards the reordered search documents to the client device. Similar to the PC-based search, the document titles are listed in the display. The user could select one of the search documents to seek further details. The request for selected document is sent to server in XML. An example would be:

```
<request>document</request><value>document_id</value>
```

Server query S(q): Context-aware search manager inspects the user context profiles and generates the augmented profiles. The user query is merged with the augmented profiles to solve the ambiguous query. The modified query is forwarded to the Yahoo search engine via the search engine URL.

Yahoo Documents Y(d): Yahoo search takes the server query S(q) and solves the query. This results in a set of documents which is sent back to the context-aware search manager at the server side. The document set are presented in XML as follows:

```
<ResultSet …>
<Result>
<Title>Apple Inc.</Title>
<Summary>
Macintosh hardware, software, and Internet tools. Offering Quicktime info, developer resources, and other items related to Apple computers.
</Summary>
<Url>http://www.apple.com/</Url>
<ClickUrl>http://www.apple.com/</ClickUrl>
<DisplayUrl>www.apple.com/</DisplayUrl>
<ModificationDate>1235548800</ModificationDate>
<MimeType>text/html</MimeType>
```

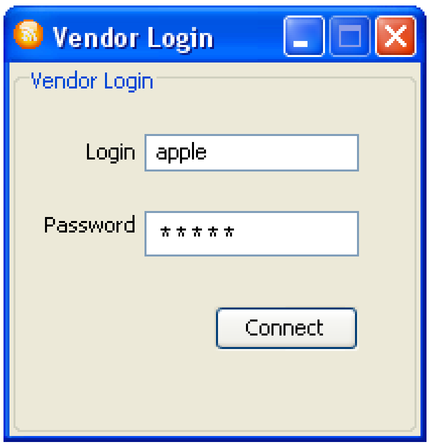
```
</Result>
…
<Result>
<Title>
Apple - Download music and more with iTunes. Play it all on iPod.
</Title>
<Summary>
Apple's iTunes allows both Mac and Windows users to sync their iPods and download music, movies, and games as well as encode, organize, and play their own music library.
</Summary>
<Url>http://www.apple.com/itunes</Url>
<ClickUrl>http://www.apple.com/itunes</ClickUrl>
<DisplayUrl>www.apple.com/itunes</DisplayUrl>
<ModificationDate>1235635200</ModificationDate>
<MimeType>text/html</MimeType>
</Result>
</ResultSet>
```

Server Reordered Document: S(d): Context-aware search manager receives the document set from Yahoo search engine. It parses the document set in XML and reorders the document set with respect to the user context. Document which is more related to current user profiles is ranked in higher order. High ranking documents appear first in the reordered document set. In order to reduce the download time, only document ids and document titles are sent to the client device.

<u>Server Response s(d):</u> When a user chooses to select one document for additional detail, user sends a request, u(d), to the search manager at the server side. Search manager first links user selection to the user query and saves such information in the database. This linkage is acknowledged as the best source to solve the query. The selected document(s) will be considered by search manager to solve the specified user query, u(q), in the future. Such information will affect the document reorder as well. At the end, the server fetches the document detail from memory/database and forwards the content to the mobile device.

5.3 Vendor GUI

The Vendor GUI component has a login frame as shown in Figure 5.2 Vendor Login Frame


Vendor Login

Vendor Login

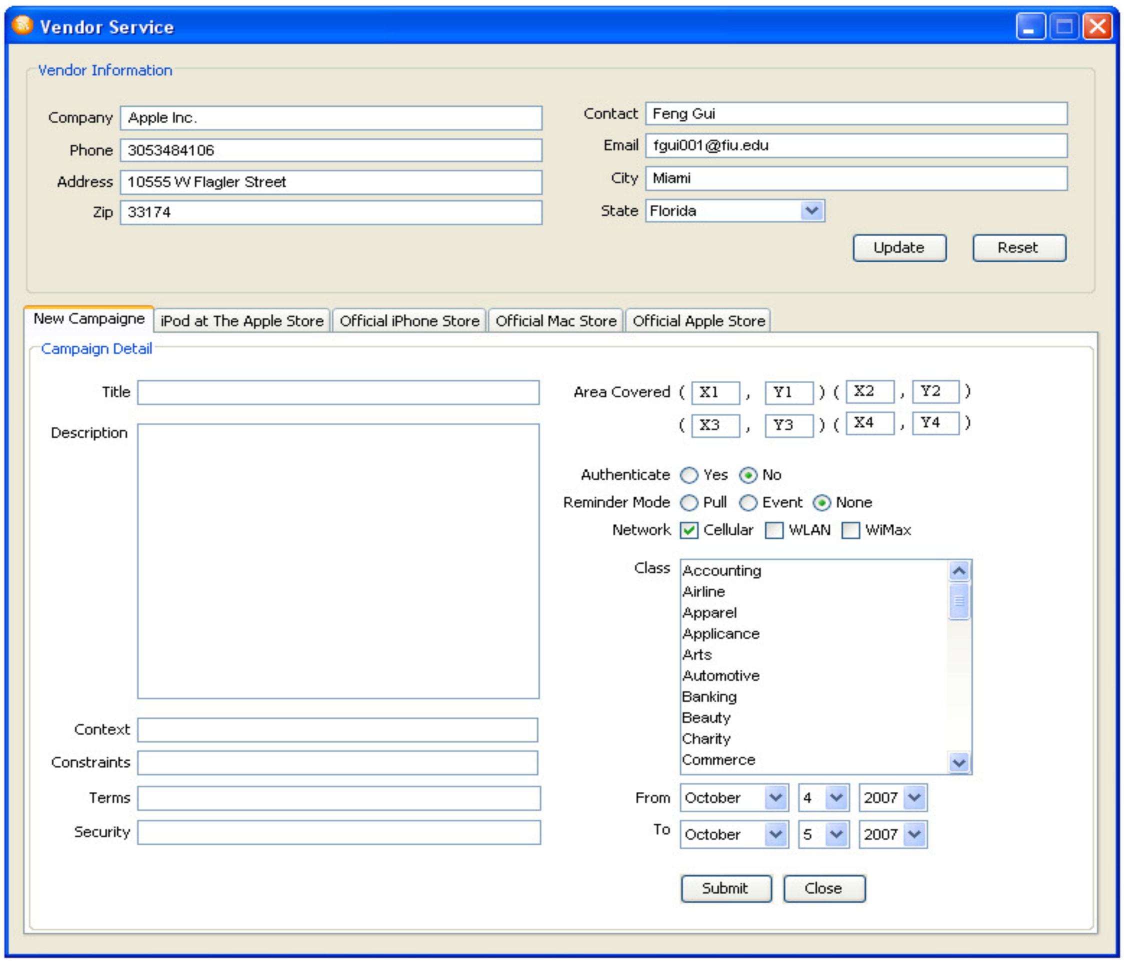
Login apple

Password *****

Figure 5.2 Vendor Login Frame

The login session identifies the vendor against the database record. MySQL stores the vendor's login information in this case. The vendor service registration frame is loaded after vendor successfully login on to the carrier's network.

The vendor service registration shown in Figure 5.3 requires the information about the service vendor and what it provides to mobile users. This information includes the area under which the service is available, service description, service type, and policy, among others.


Vendor Service

Vendor Information

Company	Apple Inc.	Contact	Feng Gui
Phone	3053484106	Email	fgui001@fiu.edu
Address	10555 W Flagler Street	City	Miami
Zip	33174	State	Florida

Update Reset

New Campaigne | iPod at The Apple Store | Official iPhone Store | Official Mac Store | Official Apple Store

Campaign Detail

Title

Description

Context

Constraints

Terms

Security

Area Covered (X1 , Y1) (X2 , Y2)
(X3 , Y3) (X4 , Y4)

Authenticate Yes [✓ No]

Reminder Mode Pull Event [✓ None]

Network [✓ Cellular] WLAN WiMax

Class: Accounting, Airline, Apparel, Applicance, Arts, Automotive, Banking, Beauty, Charity, Commerce

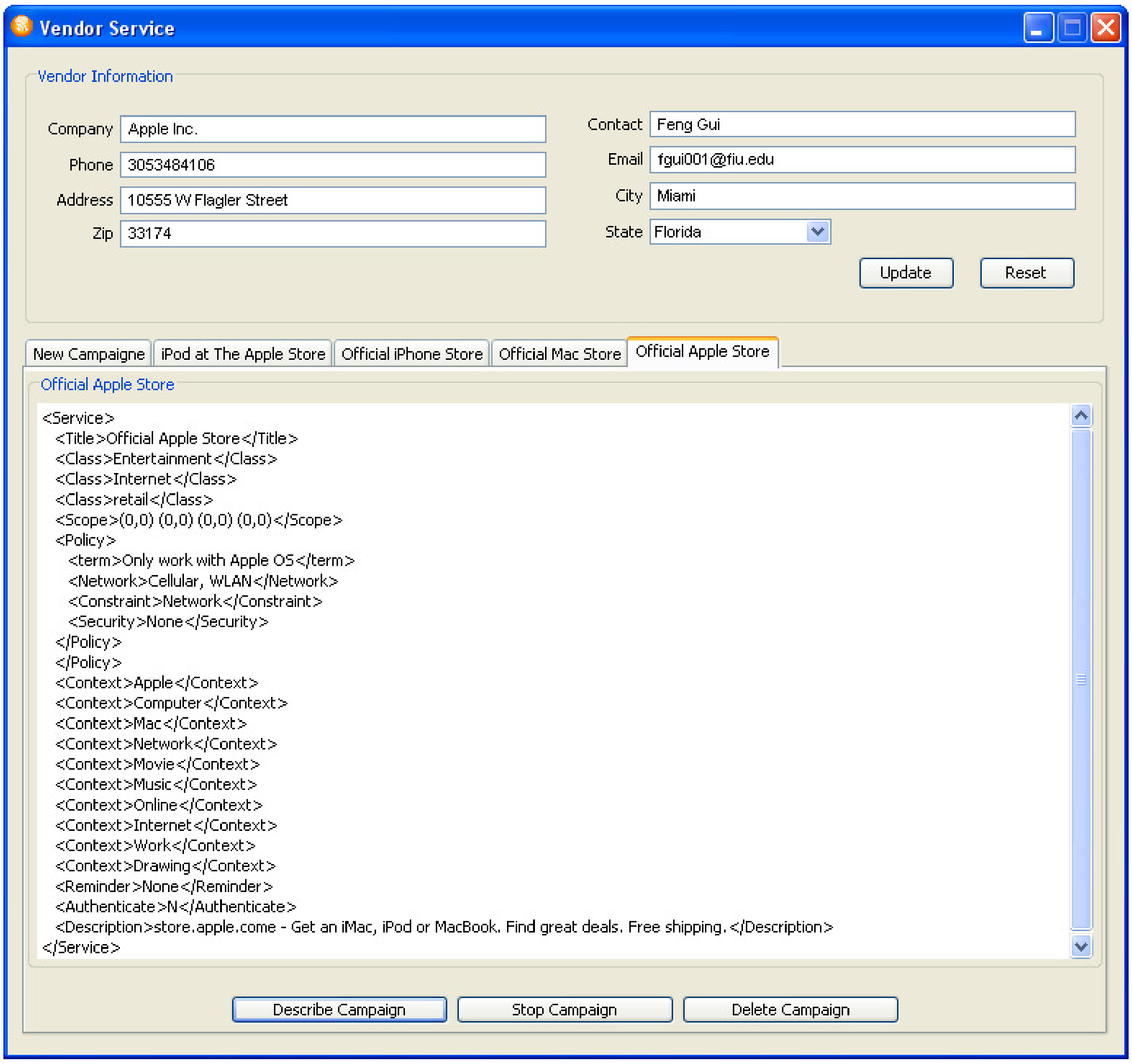
From October 4 2007

To October 5 2007

Submit Close

Figure 5.3 Vendor Service Registration

After the vendor registers the service successfully as shown in Figure 5.4, service registry compiles vendor information into an XML file. Context manager further stores information into MySQL.



Company	Apple Inc.	Contact	Feng Gui
Phone	3053484106	Email	fgui001@fiu.edu
Address	10555 W Flagler Street	City	Miami
Zip	33174	State	Florida

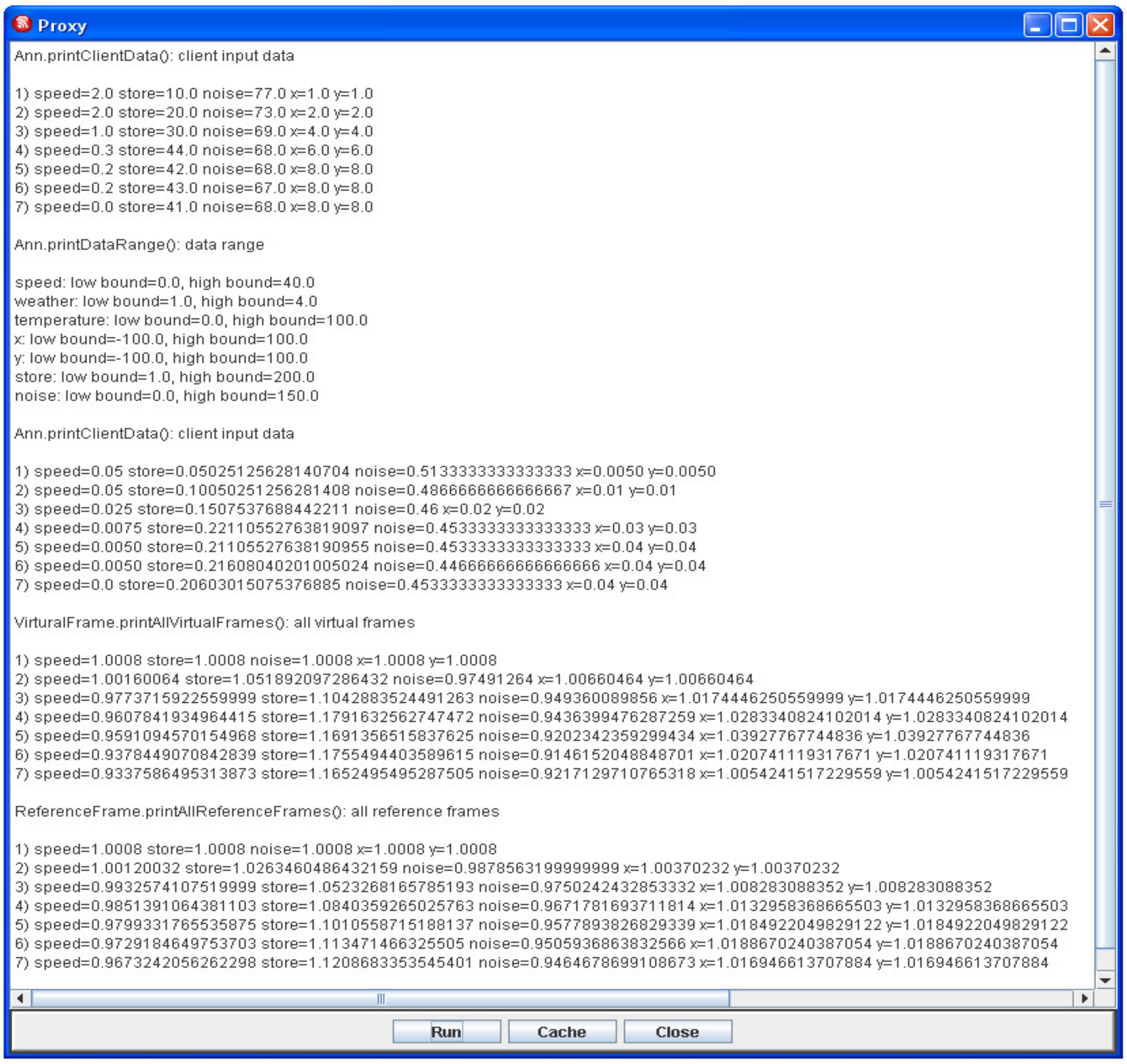
```
<Service>
  <Title>Official Apple Store</Title>
  <Class>Entertainment</Class>
  <Class>Internet</Class>
  <Class>retail</Class>
  <Scope>(0,0) (0,0) (0,0) (0,0)</Scope>
  <Policy>
    <term>Only work with Apple OS</term>
    <Network>Cellular, WLAN</Network>
    <Constraint>Network</Constraint>
    <Security>None</Security>
  </Policy>
  </Policy>
  <Context>Apple</Context>
  <Context>Computer</Context>
  <Context>Mac</Context>
  <Context>Network</Context>
  <Context>Movie</Context>
  <Context>Music</Context>
  <Context>Online</Context>
  <Context>Internet</Context>
  <Context>Work</Context>
  <Context>Drawing</Context>
  <Reminder>None</Reminder>
  <Authenticate>N</Authenticate>
  <Description>store.apple.come - Get an iMac, iPod or MacBook. Find great deals. Free shipping.</Description>
</Service>
```

Figure 5.4. Display Service Registered in Text

5.4 Evaluation Method

In this method, two user scenarios are provided to evaluate the context-aware designs at both server and client sides. These scenarios are close to the daily activities of mobile users. The value of μ, the machine specific coefficient in Eq. (4.6), is set between 0.0008 and 0.005 in the simulation process. To simplify the context computing, it is necessary to quantify some context entities into digital format. For example, *weather*: sunny = 1, cloudy = 2, raining = 3; *Location* is expressed by (x, y) such as (1, 1), (2, 2), …, (m, n); numbers are assigned arbitrarily to *store type*: department store 10, Woman Cloth = 20, foot wear = 30; *food store* has a type 40 and up, specifically Chinese food counter = 41, Japanese food counter = 42, Subway = 43, Pizza Hut = 44, and so on.

To simplify the simulation, only a few context entities are presented in each simulation. Figure 5.5 shows the simulation of the proxy on the client device. Proxy reads client situation and compiles virtual frames and reference frames for the user profiles. Proxy is used to simulate the scenarios in the following sections.



```
Ann.printClientData(): client input data

1) speed=2.0 store=10.0 noise=77.0 x=1.0 y=1.0
2) speed=2.0 store=20.0 noise=73.0 x=2.0 y=2.0
3) speed=1.0 store=30.0 noise=69.0 x=4.0 y=4.0
4) speed=0.3 store=44.0 noise=68.0 x=6.0 y=6.0
5) speed=0.2 store=42.0 noise=68.0 x=8.0 y=8.0
6) speed=0.2 store=43.0 noise=67.0 x=8.0 y=8.0
7) speed=0.0 store=41.0 noise=68.0 x=8.0 y=8.0

Ann.printDataRange(): data range

speed: low bound=0.0, high bound=40.0
weather: low bound=1.0, high bound=4.0
temperature: low bound=0.0, high bound=100.0
x: low bound=-100.0, high bound=100.0
y: low bound=-100.0, high bound=100.0
store: low bound=1.0, high bound=200.0
noise: low bound=0.0, high bound=150.0

Ann.printClientData(): client input data

1) speed=0.05 store=0.050251256281407004 noise=0.5133333333333333 x=0.0050 y=0.0050
2) speed=0.05 store=0.10050251256281408 noise=0.4866666666666667 x=0.01 y=0.01
3) speed=0.025 store=0.1507537688442211 noise=0.46 x=0.02 y=0.02
4) speed=0.0075 store=0.22110552763819097 noise=0.4533333333333333 x=0.03 y=0.03
5) speed=0.0050 store=0.21105527638190955 noise=0.4533333333333333 x=0.04 y=0.04
6) speed=0.0050 store=0.21608040201005024 noise=0.44666666666666666 x=0.04 y=0.04
7) speed=0.0 store=0.20603015075376885 noise=0.4533333333333333 x=0.04 y=0.04

VirturalFrame.printAllVirtualFrames(): all virtual frames

1) speed=1.0008 store=1.0008 noise=1.0008 x=1.0008 y=1.0008
2) speed=1.00160064 store=1.051892097286432 noise=0.97491264 x=1.00660464 y=1.00660464
3) speed=0.9773715922559999 store=1.1042883524491263 noise=0.949360089856 x=1.0174446250559999 y=1.0174446250559999
4) speed=0.9607841934964415 store=1.1791632562747472 noise=0.9436399476287259 x=1.0283340824102014 y=1.0283340824102014
5) speed=0.9591094570154968 store=1.1691356515837625 noise=0.9202342359299434 x=1.03927767744836 y=1.03927767744836
6) speed=0.9378449070842839 store=1.1755494403589615 noise=0.9146152048848701 x=1.020741119317671 y=1.020741119317671
7) speed=0.9337586495313873 store=1.1652495495287505 noise=0.9217129710765318 x=1.0054241517229559 y=1.0054241517229559

ReferenceFrame.printAllReferenceFrames(): all reference frames

1) speed=1.0008 store=1.0008 noise=1.0008 x=1.0008 y=1.0008
2) speed=1.00120032 store=1.0263460486432159 noise=0.9878563199999999 x=1.00370232 y=1.00370232
3) speed=0.9932574107519999 store=1.0523268165785193 noise=0.9750242432853332 x=1.008283088352 y=1.008283088352
4) speed=0.9851391064381103 store=1.0840359265025763 noise=0.9671781693711814 x=1.0132958368665503 y=1.0132958368665503
5) speed=0.9799331765535875 store=1.1010558715188137 noise=0.9577893826829339 x=1.0184922049829122 y=1.0184922049829122
6) speed=0.9729184649753703 store=1.113471466325505 noise=0.9505936863832566 x=1.0188670240387054 y=1.0188670240387054
7) speed=0.9673242056262298 store=1.1208683353545401 noise=0.9464678699108673 x=1.016946613707884 y=1.016946613707884
```

Figure 5.5. Proxy on Client

5.5 Scenario One: Mobile user roaming in the street

In scenario one, the user carrying mobile device walks at varying speeds in a fixed direction. During his trip it rains. User runs and stops at a gas station, waits for the rain to stop. Seven steps describe this first scenario:

- The user walks by location (1, 1) at the speed of 3 mile/hr when the weather is sunny (1), and the temperature is 77. He goes by Burger King.

- While the user keeps walking at the same speed passing location (2, 2), the weather changes to cloudy (2) and the temperature drops to 73. He passes by a women shoes store.

- When it starts raining (3) and temperature continues to drop to 69, the user is running at the speed of 6 mile/hr towards the same direction and passing location (4, 4). He goes by a pawn shop.

- User is still running, passing location (6, 6). Temperature changes slightly to 68. It is raining (3). He walks by a body shop.

- User keeps running at the same speed reaching location (8, 8) while it is raining (3). Temperature is unchanged. He passes a department store.

- User stops at location (8, 8) while temperature drops slightly to 67 and it is raining (3). He goes by a Chinese Restaurant.

- User stays at location (8, 8) while temperature rises slightly to 68 and it is raining (3). He is at Shell gas station. The above 7 steps are summarized in Table 5.1.

Table 5.1 Context entities of scenario 1

Step	Speed	Weather	Temperature	x	y
S(1)	3mile/hr	1(sunny)	77	1	1
S(2)	3mile/hr	2(cloudy)	73	2	2
S(3)	6mile/hr	3(raining)	69	4	4
S(4)	6mile/hr	3(raining)	68	6	6
S(5)	6mile/hr	3(raining)	68	8	8
S(6)	0mile/hr	3(raining)	67	8	8
S(7)	0mile/hr	3(raining)	68	8	8

Simulation results for scenario 1 are given in Table 5.1, whose first column from the left represents the virtual frames calculated at different time instances. The other columns store the weight values for the context entities. The proxy updates weights as it adapts to context input. As time goes by, the virtual frames reflect the change in the speed. The weather changes dramatically in the first three steps. The weight rises to reflect weather changes. It drops as it continues to rain. Also we observe the weight in x and y drops as the user stops. The simulation results are listed in Tables 5.2 and 5.3 for the virtual and reference frames. The reference frames listed in Table 5.3 are less dramatic compared to virtual frames. This is expected as the reference frame is the moving average of the virtual frames. In Table 5.4, the network returns content pertaining to the current user context entities.

Table 5.2 Virtual frames of scenario 1

frame(t)	Speed	Weather	Temperature	x	Y
f(1)	1.001	1.001	1.001	1.001	1.001
f(2)	1.001	1.335	0.961	1.006	1.006
f(3)	1.077	1.724	0.922	1.017	1.017
f(4)	1.026	1.274	0.913	1.027	1.027
f(5)	0.988	0.941	0.878	1.038	1.038
f(6)	0.836	0.691	0.869	1.019	1.019
f(7)	0.737	0.498	0.879	1.003	1.003

Table 5.3 Reference frames of scenario 1

frame(t)	Speed	Weather	Temperature	x	Y
f(1)	1.001	1.001	1.001	1.001	1.001
f(2)	1.001	1.168	0.981	1.003	1.003
f(3)	1.026	1.353	0.961	1.008	1.008
f(4)	1.026	1.333	0.949	1.013	1.013
f(5)	1.018	1.255	0.935	1.018	1.018
f(6)	0.988	1.161	0.924	1.018	1.018
f(7)	0.952	1.066	0.918	1.016	1.016

Table 5.4 Context Result for scenario 1

Source	Description
weather.com	National Forecast, free local weather alerts
Yahoo! weather	Yahoo weather forecasts, resources, and categories.
Local.com	Free live weather forecast
Shell	Locate Shell gas station
BP Gas	Great deals on gas
Exxon	Speedpass simplifies your life
MapQuest	Maps, directions, and more
Yahoo! Maps	Finding your way has never been easier

The context search engine organizes and merges data based on context weights. Since weather and location context have higher weights in the context profile, in the reference frame we see the top returned search results. The stay at a nearby gas station increases the location-dependent information such as service at gas station and driving directions. It is intuitive that user selection on this set of data would further feedback the device proxy and hikes the context weights related to the selection.

5.6 Scenario Two: Roaming off the main campus of Florida International University

In scenario two, a mobile user goes off the main campus of Florida International University. The Google Earth screen shot of this travel route is as shown in Figure 5.6.

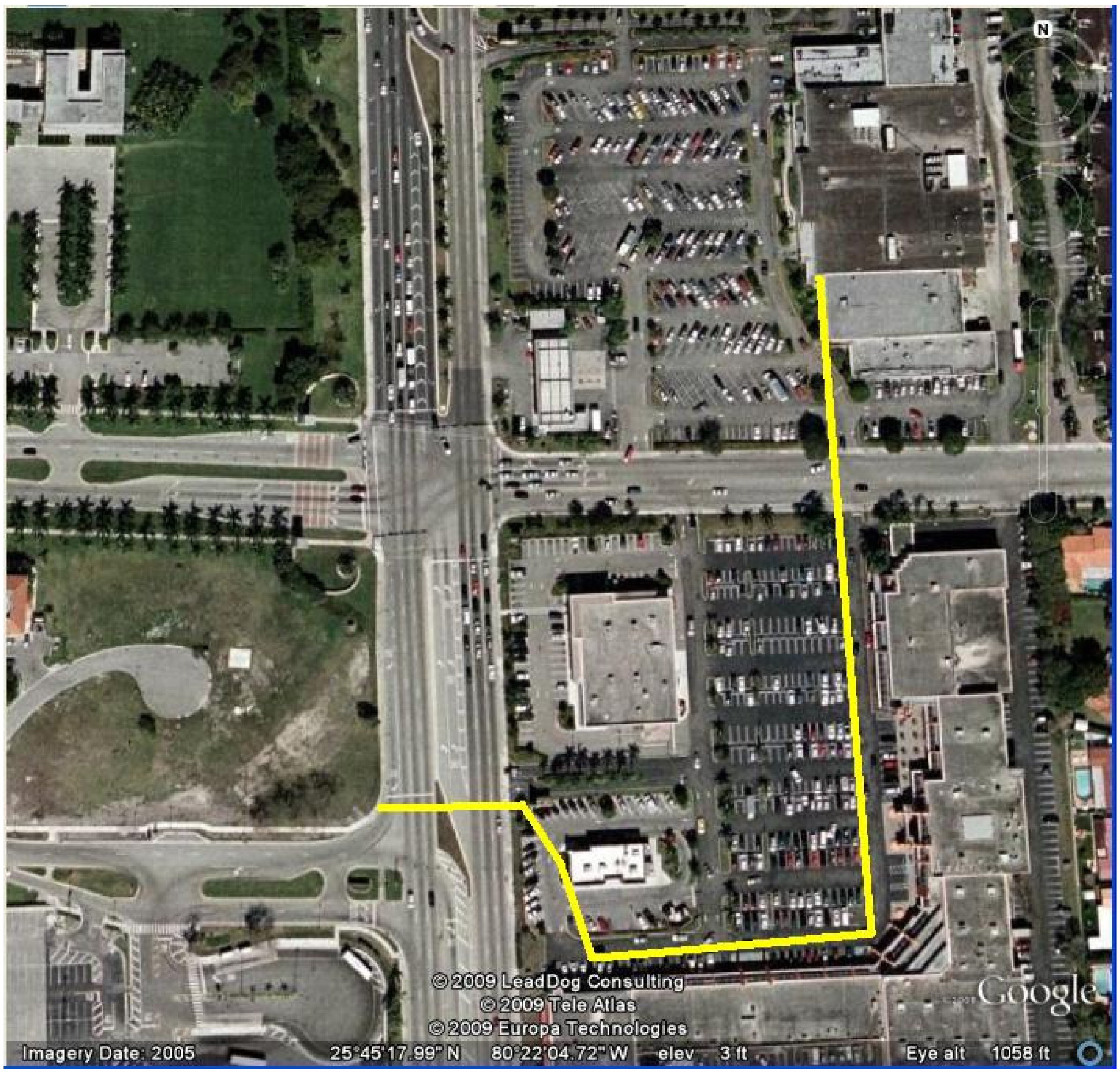


Figure 5.6 Google Earth Screen Shot of Travel Route for Scenario 2

The user in this scenario walks east bound. We describe user activities through the following steps.

- User is walking off the main campus and crossing the 107th Avenue. The location coordinate (6, 6) at the speed of 2 mile/hr while the noise level is at 60 db.
- User is still walking at the same speed passing Wendy's restaurant and location coordinate (4, 7). The noise level rises to 63 db.
- The noise level remains the same. User walks by Subway, location (4, 8), and at the speed of 1 mile/hr.
- User walks at 0.5 mile/hr and passes Gold's Gym. The noise level continues to rise to 77 db. Gold's Gym has location coordinate (2, 6).
- User walks by Terry Bank at 0.2 mile/hr. The noise level is at 67 db at this regional bank with coordinate (2, 5).
- User arrives at Little Caesars. User walks by at this restaurant whose location coordinator is (2, 4) with a noise level of 67db.
- User reaches Publix (2, 3) and starts using his mobile device to search information. Noise level rises to 67 db. The above 7 steps are summarized in Table 5.5.

Table 5.5 Context entities of scenario 2

Step	Speed	Store	Noise	x	y
S(1)	2mile/h	59	60	6	6
S(2)	2mile/h	61	63	4	7
S(3)	1mile/h	62	63	4	8
S(4)	0.5mile/h	82	77	2	6
S(5)	0.2mile/h	75	67	2	5
S(6)	0.2mile/h	67	67	2	4
S(7)	0mile/h	68	67	2	3

The simulation results are listed in Tables 5.6 and 5.7 for the virtual and reference frames.

Table 5.6. Virtual frames of scenario 2

frame(t)	Speed	Place	Noise	x	y
f(1)	1.000	1.000	1.000	1.000	1.000
f(2)	1.005	1.038	1.055	1.338	1.172
f(3)	1.512	1.078	0.005	0.452	1.345
f(4)	2.276	1.408	0.006	0.681	1.687
f(5)	3.653	1.536	0.007	-0.405	1.977
f(6)	0.809	1.707	0.001	0.133	2.383
f(7)	1.624	1.742	0.0004	-0.018	2.991

Table 5.7 Reference frames of scenario 2

frame(t)	Speed	Place	Noise	X	y
f(1)	1.003	1.019	1.027	1.169	1.086
f(2)	1.172	1.039	0.686	0.930	1.172
f(3)	1.448	1.131	0.516	0.868	1.301
f(4)	1.889	1.212	0.414	0.613	1.436
f(5)	1.710	1.295	0.346	0.533	1.594
f(6)	1.697	1.359	0.297	0.454	1.794

In this scenario, the context weights in the virtual frames capture the context changes detected. As the context entities change is less obvious, the context weights decrease. The weight for location changes significantly as user goes by different places. The location is an important context entity for this user. When it comes down to the ambiguous query, the context entities help the search engine to identify the user intention.

In Table 5.8, the network returns content pertaining to the current user context entities. The context-awareness server is able to sense that the mobile user is just roaming off FIU campus. It further provides information based on the current user situation without the user input or query.

Table 5.8 Context Result for scenario 2

Source	Description
Yahoo! Maps	Finding your way has never been easier
MapQuest	Direction beyond point
Walmart	Apparel, baby, electronics
JCPenny	Everyday matters.
Yahoo Weather	Your local weather channel
Wendy's	Supper value menus.
Tropical Chinese	Authentic cuisine, real deal.

5.7 Search Engine Manager

The context search manager personalizes the user query by taking user profiles into account. Intuitively, context profile should provide additional information which could assist the search engine to fetch documents related to user current situation. For example the user profile might provide location information and history data. The proposed user feed back system provides a list of documents which are selected by other users when they view the files based on the same search key word. Currently, the user suggested documents are merged with the documents based on context search. If the mobile user chooses to review any one of the documents, the selection will be reflected in the server database. The internal tally for this particular document is incremented in response to the key word. If more users keep selecting the same document with regards to the search word, the selected document will move up in the rank for the future reference.

Yahoo search API is utilized to solve the user queries. Even though Google search engine is the most popular one, Yahoo search engine is second in terms of search results. Yahoo search engine provides free API to programmers, which makes this type of simulation possible with real time search results. The simulation results are thus tightly dependent on the accuracy of the Yahoo search algorithm. However, the user feedback is implemented on the server side to further improve the search accuracy.

Figure 5.7 shows the simple GUI for testing the context search engine. The user's search key word is sent to the search engine. If the user clicks the "Send" button, the search engine will provide related documents which might be related to user profiles.

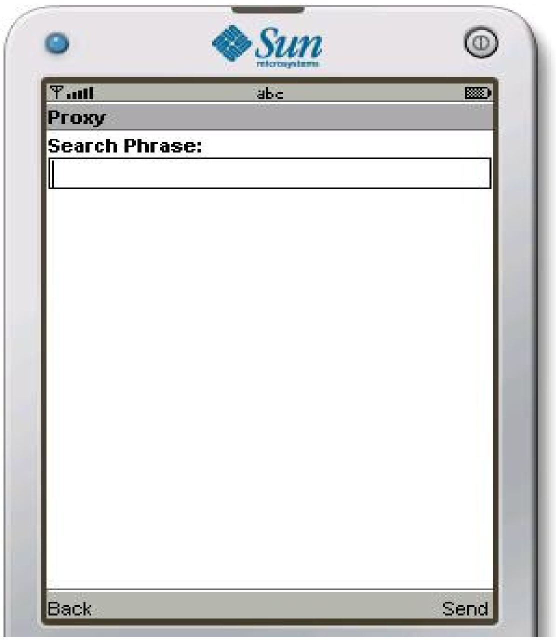
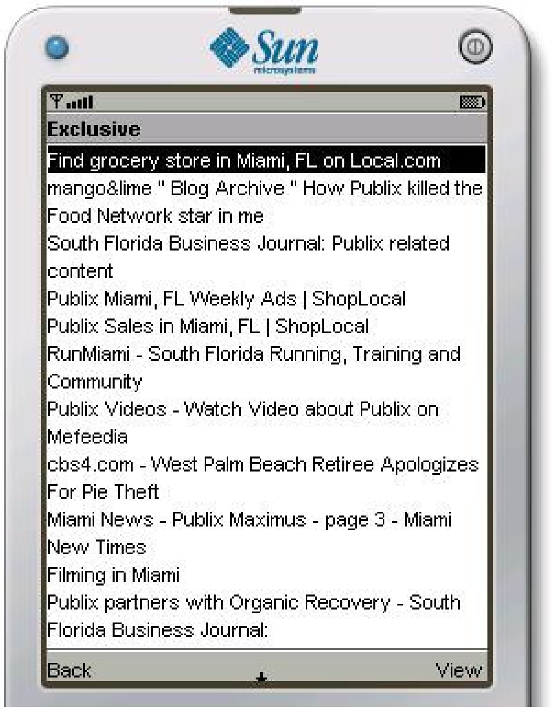


Figure 5.7 Context Search Application

Recall in scenario two in previous section, the generated user reference frame at step 7 is listed as follows:

frame(t)	Speed	Place	Noise	x	y
f(6)	1.697	1.359	0.297	0.454	1.794

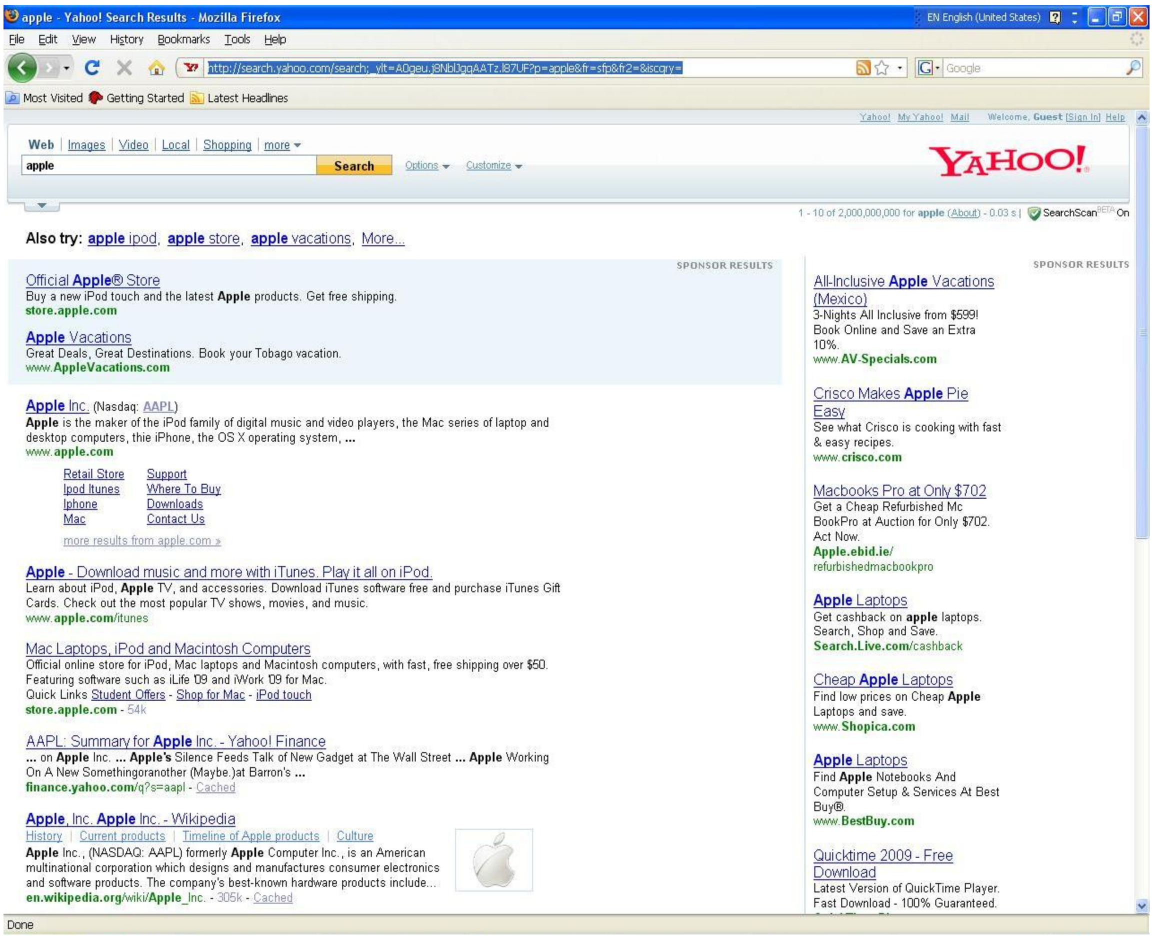
The user reference frame contains location information and store information that will help the search engine to optimize the search results. Suppose that the user enters the key word “apple”. The interesting fact about “apple” is that this word is frequently associated with the high-tech company Apple Inc. which sells hot gadgets such as iPods and iPhones. In general, most search engines including Google will consider that the user was searching for products from Apple Inc. On the other hand, apple is a fruit which is a type of food. Figure 5.8 shows the returned documents based on the user profile only.



Find grocery store in Miami, FL on Local.com
mango&lime " Blog Archive " How Publix killed the Food Network star in me
South Florida Business Journal: Publix related content
Publix Miami, FL Weekly Ads | ShopLocal
Publix Sales in Miami, FL | ShopLocal
RunMiami - South Florida Running, Training and Community
Publix Videos - Watch Video about Publix on Mefeedia
cbs4.com - West Palm Beach Retiree Apologizes For Pie Theft
Miami News - Publix Maximus - page 3 - Miami New Times
Filming in Miami
Publix partners with Organic Recovery - South Florida Business Journal:

Figure 5.8 Returned Search Result for Scenario 1

Because the context file records that the mobile user is near the restaurant and grocery store, so the search word “apple” is more likely related to food. The search engine returns information about grocery stores and restaurants. On the other hand, the context search manager also considers the possibility that the search key word is related to products from Apple Inc. However, user needs to scroll down to see documents related to apple products.


apple - Yahoo! Search Results - Mozilla Firefox
File Edit View History Bookmarks Tools Help
http://search.yahoo.com/search;_ylt=A0geu.j8NbIJgqAATz.l87UF?p=apple&fr=sfp&fr2=&iscqry=
Most Visited Getting Started Latest Headlines
Web | Images | Video | Local | Shopping | more
apple
Search
Options
Customize
Done

1 - 10 of 2,000,000,000 for **apple** (About) - 0.03 s | SearchScan BETA On

Also try: **apple** ipod, **apple** store, **apple** vacations, More...

SPONSOR RESULTS

Official **Apple**® Store
Buy a new iPod touch and the latest **Apple** products. Get free shipping.
store.apple.com

Apple Vacations
Great Deals, Great Destinations. Book your Tobago vacation.
www.**AppleVacations.com**

Apple Inc. (Nasdaq: AAPL)
Apple is the maker of the iPod family of digital music and video players, the Mac series of laptop and desktop computers, thie iPhone, the OS X operating system, **...**
www.**apple.com**

Retail Store | Support
Ipod Itunes | Where To Buy
Iphone | Downloads
Mac | Contact Us

more results from apple.com »

Apple - Download music and more with iTunes. Play it all on iPod.
Learn about iPod, **Apple** TV, and accessories. Download iTunes software free and purchase iTunes Gift Cards. Check out the most popular TV shows, movies, and music.
www.**apple.com**/itunes

Mac Laptops, iPod and Macintosh Computers
Official online store for iPod, Mac laptops and Macintosh computers, with fast, free shipping over $50. Featuring software such as iLife '09 and iWork '09 for Mac.
Quick Links Student Offers - Shop for Mac - iPod touch
store.apple.com - 54k

AAPL: Summary for **Apple** Inc. - Yahoo! Finance
... on **Apple** Inc. **... Apple's** Silence Feeds Talk of New Gadget at The Wall Street **... Apple** Working On A New Somethingoranother (Maybe.)at Barron's **...**
finance.yahoo.com/q?s=aapl - Cached

Apple, Inc. **Apple** Inc. - Wikipedia
History | Current products | Timeline of Apple products | Culture
Apple Inc., (NASDAQ: AAPL) formerly **Apple** Computer Inc., is an American multinational corporation which designs and manufactures consumer electronics and software products. The company's best-known hardware products include...
en.wikipedia.org/wiki/**Apple**_Inc. - 305k - Cached

SPONSOR RESULTS

All-Inclusive **Apple** Vacations (Mexico)
3-Nights All Inclusive from $599! Book Online and Save an Extra 10%.
www.**AV-Specials.com**

Crisco Makes **Apple** Pie Easy
See what Crisco is cooking with fast & easy recipes.
www.**crisco.com**

Macbooks Pro at Only $702
Get a Cheap Refurbished Mc BookPro at Auction for Only $702. Act Now.
Apple.ebid.ie/
refurbishedmacbookpro

Apple Laptops
Get cashback on **apple** laptops. Search, Shop and Save.
Search.Live.com/cashback

Cheap **Apple** Laptops
Find low prices on Cheap **Apple** Laptops and save.
www.**Shopica.com**

Apple Laptops
Find **Apple** Notebooks And Computer Setup & Services At Best Buy®.
www.**BestBuy.com**

Quicktime 2009 - Free Download
Latest Version of QuickTime Player. Fast Download - 100% Guaranteed.

Figure 5.9 Yahoo Search Results without User Context

In comparison, the Yahoo search engine, in general, is not aware of the user context. If the user submits the query "apple" to the search engine, the search engine returns the result documents that are mostly related to the Apple INC. This is due to the popularity of the Apple products such as IPhone and IPod. Clearly the mobile user benefits from the context-aware search approach.

In another test, scenario 3, the mobile user is situated in the Dolphin Mall with the route as shown in the layout of Figure 5.10.

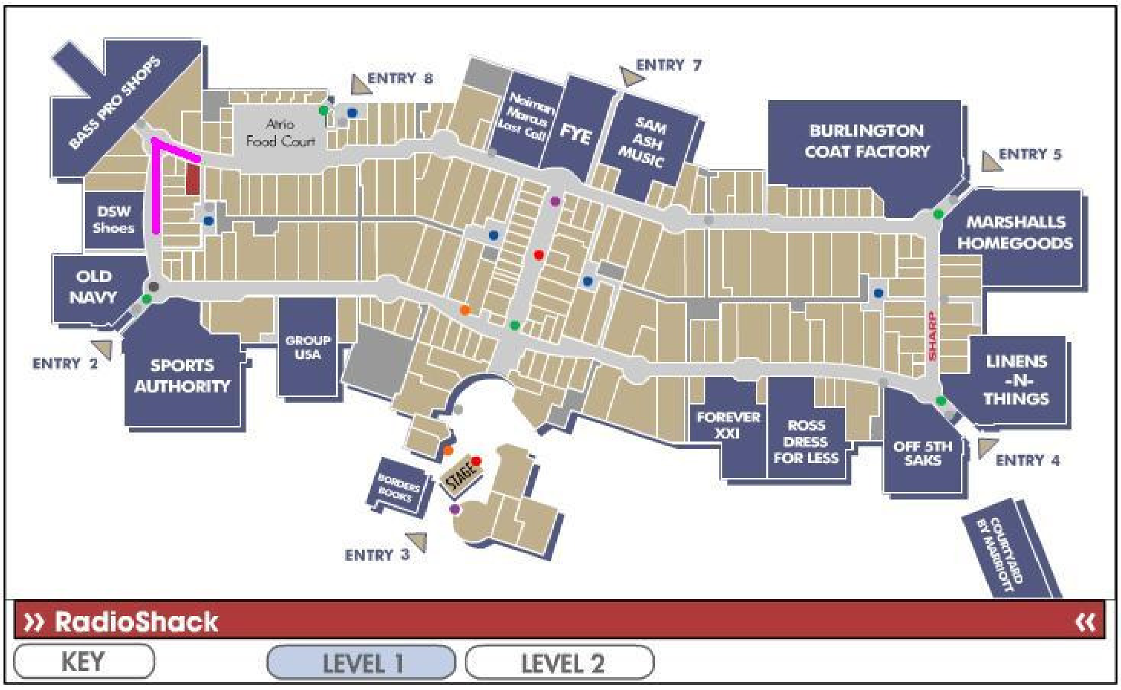


Figure 5.10 Scenario for a Travel Route in the Dolphin Mall

In this scenario, the following stores are in the traveled route, Kitchen Collection, Player's Closet, Sports Fan, Men's Wear, International Electronics Super Store, World Time, and Radio Shack.

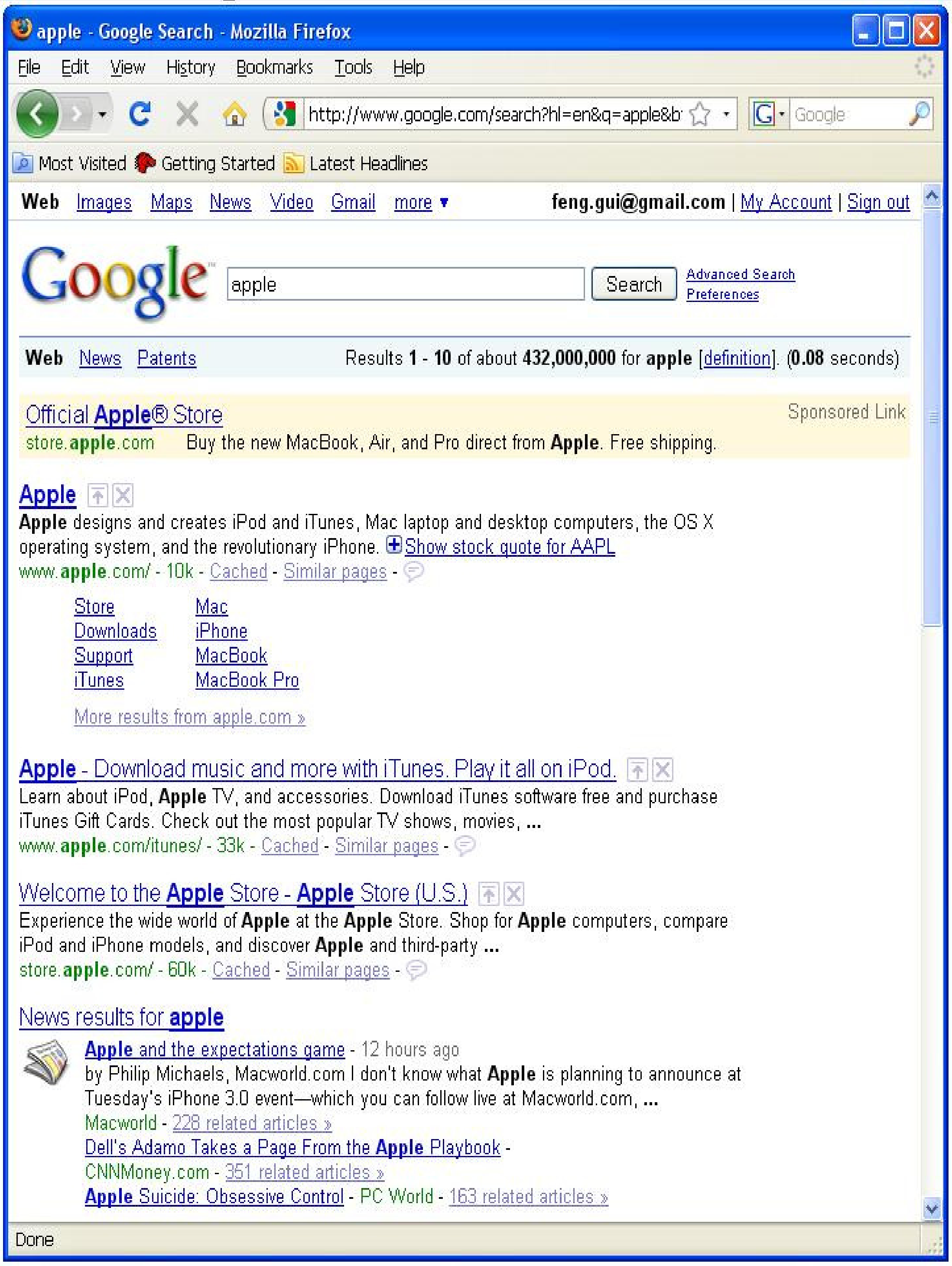
Again, the user search for the same word "apple". Figure 5.11 shows the search result from Google. Clearly, Google is not aware of the user local context. It basically returns general search results that are popular but with no coverage for service from local companies. In comparison, the context manager focuses on the user context and tries to bring the local service and products to the mobile user. Figure 5.12 shows the document lists generated by the context search engine. More returned documents are related to the iPod, Apple computer, or local companies that offer apple products and services.

Table 5.9 Context entities in Dolphin Mall

Step	Speed	Store	Noise	x	y
S(1)	2mile/h	200	60	1	1
S(2)	2mile/h	185	63	1	2
S(3)	1mile/h	33	63	1	3
S(4)	0.7mile/h	55	69	1	4
S(5)	0.5mile/h	95	68	1	5
S(6)	0.4mile/h	105	68	2	5
S(7)	0.4mile/h	99	66	3	5

Simulation results in virtual frames and reference frames. The last reference frame is derived as;

frame(t)	Speed	Place	Noise	x	y
f(6)	1.764	3.735	0.293	1.453	1.885


apple - Google Search - Mozilla Firefox

File Edit View History Bookmarks Tools Help

http://www.google.com/search?hl=en&q=apple&b

Most Visited Getting Started Latest Headlines

Web Images Maps News Video Gmail more ▾ **feng.gui@gmail.com** | My Account | Sign out

apple Search Advanced Search Preferences

Web News Patents Results **1** - **10** of about **432,000,000** for **apple** [definition]. (**0.08** seconds)

Official **Apple**® Store Sponsored Link
store.**apple**.com Buy the new MacBook, Air, and Pro direct from **Apple**. Free shipping.

Apple
Apple designs and creates iPod and iTunes, Mac laptop and desktop computers, the OS X operating system, and the revolutionary iPhone. Show stock quote for AAPL
www.**apple**.com/ - 10k - Cached - Similar pages -

Store	Mac
Downloads	iPhone
Support	MacBook
iTunes	MacBook Pro

More results from apple.com »

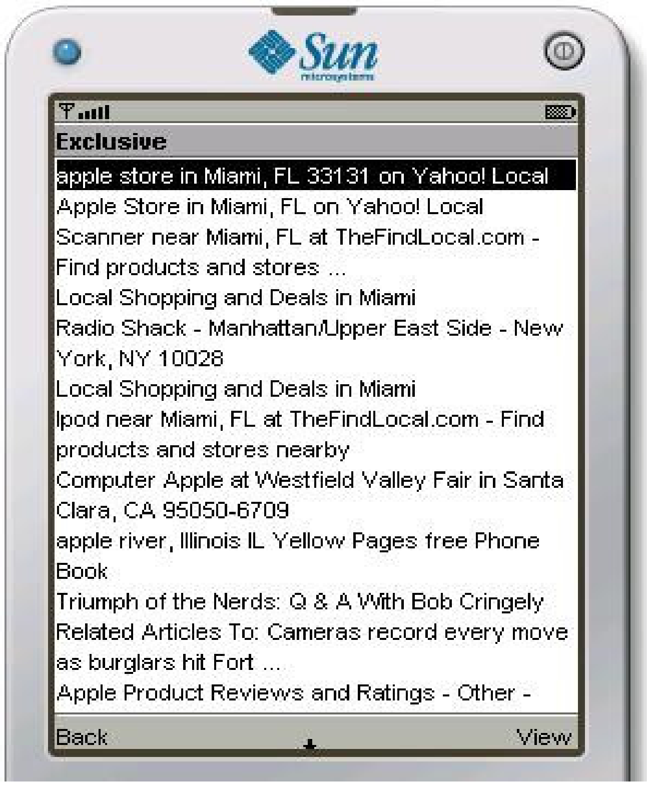
Apple - Download music and more with iTunes. Play it all on iPod.
Learn about iPod, **Apple** TV, and accessories. Download iTunes software free and purchase iTunes Gift Cards. Check out the most popular TV shows, movies, **...**
www.**apple**.com/itunes/ - 33k - Cached - Similar pages -

Welcome to the **Apple** Store - **Apple** Store (U.S.)
Experience the wide world of **Apple** at the **Apple** Store. Shop for **Apple** computers, compare iPod and iPhone models, and discover **Apple** and third-party **...**
store.**apple**.com/ - 60k - Cached - Similar pages -

News results for **apple**
Apple and the expectations game - 12 hours ago
by Philip Michaels, Macworld.com I don't know what **Apple** is planning to announce at Tuesday's iPhone 3.0 event—which you can follow live at Macworld.com, **...**
Macworld - 228 related articles »
Dell's Adamo Takes a Page From the **Apple** Playbook -
CNNMoney.com - 351 related articles »
Apple Suicide: Obsessive Control - PC World - 163 related articles »

Done

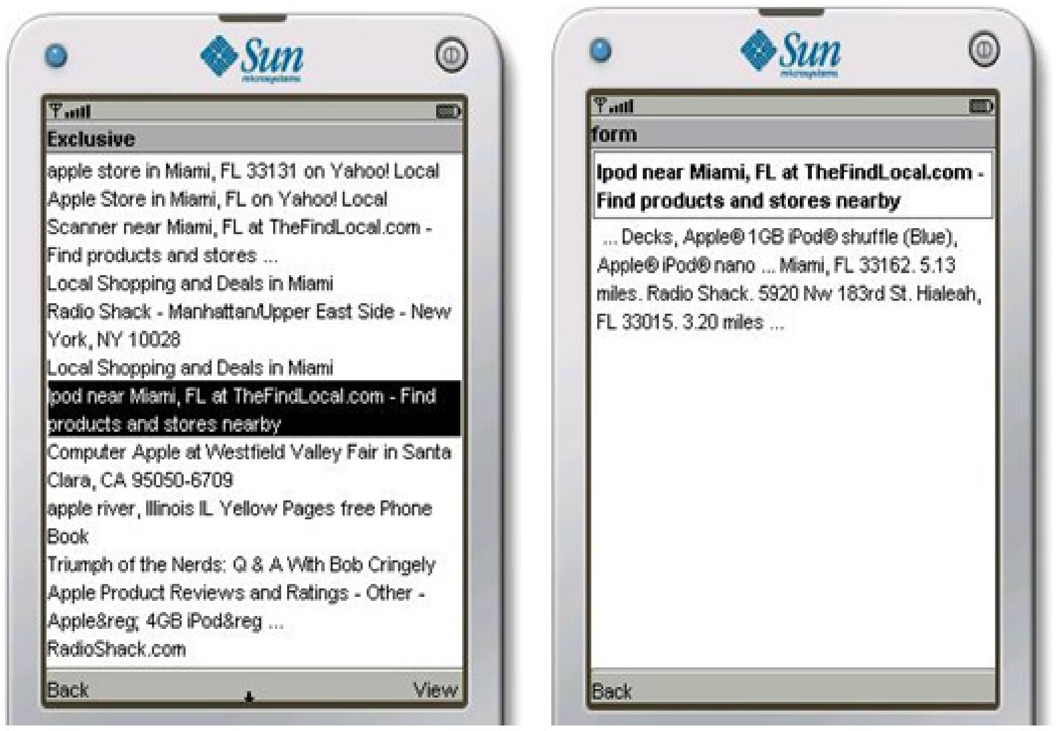
Figure 5.11 Google Search Result for Scenario 3


Exclusive

apple store in Miami, FL 33131 on Yahoo! Local
Apple Store in Miami, FL on Yahoo! Local
Scanner near Miami, FL at TheFindLocal.com - Find products and stores ...
Local Shopping and Deals in Miami
Radio Shack - Manhattan/Upper East Side - New York, NY 10028
Local Shopping and Deals in Miami
Ipod near Miami, FL at TheFindLocal.com - Find products and stores nearby
Computer Apple at Westfield Valley Fair in Santa Clara, CA 95050-6709
apple river, Illinois IL Yellow Pages free Phone Book
Triumph of the Nerds: Q & A With Bob Cringely
Related Articles To: Cameras record every move as burglars hit Fort ...
Apple Product Reviews and Ratings - Other -

Figure 5.12 Returned Search Result for Scenario 3

Context search manager implicitly processes the user selection as the feedback. The essence of the user’s feedback is that it beats any search algorithm so far. The feedback mechanism provides the semantic meaning of the searched keyword. Should the user decide to review with more detail the listed documents, the user selection contributes more votes for the selected documents. More votes in turn improve the chance to move up in the listing order for future searches. Figure 5.13 shows the user selection of one document.



Exclusive

apple store in Miami, FL 33131 on Yahoo! Local
Apple Store in Miami, FL on Yahoo! Local
Scanner near Miami, FL at TheFindLocal.com - Find products and stores ...
Local Shopping and Deals in Miami
Radio Shack - Manhattan/Upper East Side - New York, NY 10028
Local Shopping and Deals in Miami
Ipod near Miami, FL at TheFindLocal.com - Find products and stores nearby
Computer Apple at Westfield Valley Fair in Santa Clara, CA 95050-6709
apple river, Illinois IL Yellow Pages free Phone Book
Triumph of the Nerds: Q & A With Bob Cringely
Apple Product Reviews and Ratings - Other - Apple&reg; 4GB iPod&reg ...
RadioShack.com

Back View

form

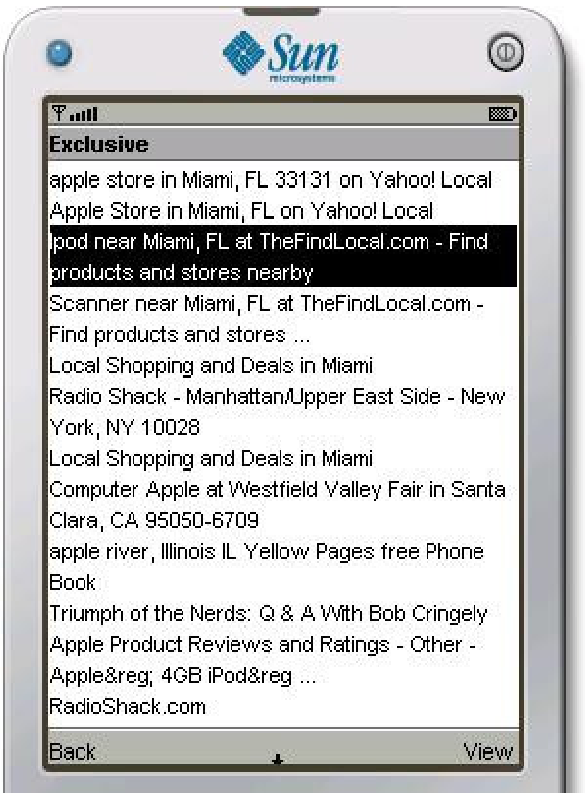
Ipod near Miami, FL at TheFindLocal.com - Find products and stores nearby

... Decks, Apple® 1GB iPod® shuffle (Blue), Apple® iPod® nano ... Miami, FL 33162. 5.13 miles. Radio Shack. 5920 Nw 183rd St. Hialeah, FL 33015. 3.20 miles ...

Back

Figure 5.13 Returned Search Result Augmented With User Feed Back

The left screen shot in figure 5.13 shows that mobile user selects the document titled "Ipod near Miami, FL at TheFindLocal.com – Find products and stores nearby" from the list of documents. The right screen shot in figure 5.13 shows the description of the document. The feed back mechanism records the user selection and relates the selected document as the high ranking document to solve the future user query. Figure 5.14 shows the new order of the search documents. The display order of the selected document is precedent.


Exclusive

apple store in Miami, FL 33131 on Yahoo! Local
Apple Store in Miami, FL on Yahoo! Local
Ipod near Miami, FL at TheFindLocal.com - Find products and stores nearby
Scanner near Miami, FL at TheFindLocal.com - Find products and stores ...
Local Shopping and Deals in Miami
Radio Shack - Manhattan/Upper East Side - New York, NY 10028
Local Shopping and Deals in Miami
Computer Apple at Westfield Valley Fair in Santa Clara, CA 95050-6709
apple river, Illinois IL Yellow Pages free Phone Book
Triumph of the Nerds: Q & A With Bob Cringely
Apple Product Reviews and Ratings - Other - Apple&reg; 4GB iPod&reg ...
RadioShack.com

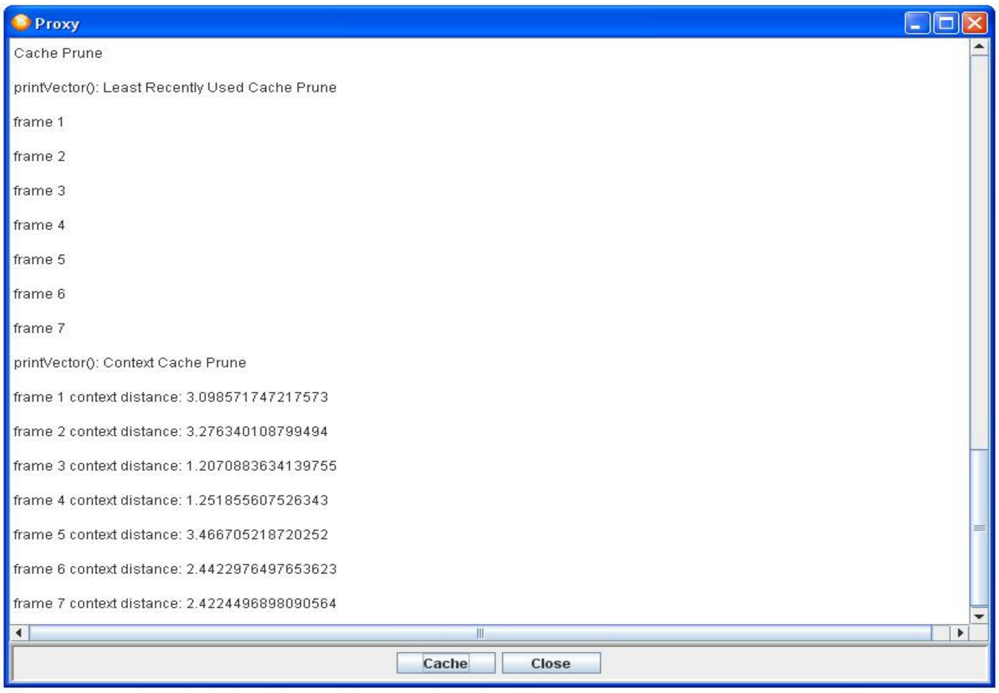
Figure 5.14 Effect of a User Vote on the Listing Order

5.8 Cache Scheme

The proposed cache stores the user data in the client device. When the user requests data that is stored in the cache, the round trip to the server is spared. Instead, the cache provides the data. Some existing cache algorithms have been proven to be effective. One of them is Least Recently Used (LRU) replacement algorithm. Due to its simplicity, LRU can be easily implemented on the mobile device. However, the LRU does not consider the user profile. The advantage for the proposed cache scheme is that it only deletes frames that are out of current user situation. Thus, it should improve the cache hit at client side. The testing case for cache scheme is built on two scenarios in previous section.

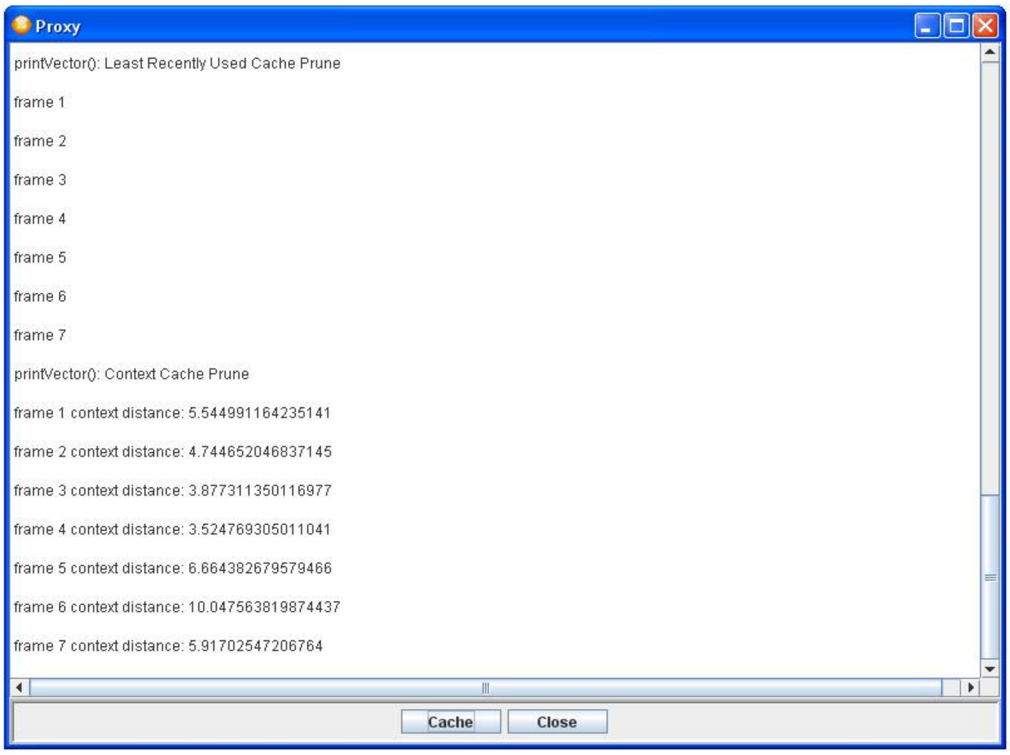
One of the scenarios in section 5.7 considers one mobile user roaming off the main FIU campus and walking around the strip mall located east of the campus. Assume the memory is full, and cache pruning must take place. With Least Recently Used (LRU) replacement algorithm, the first frame is pruned to make room for new frames. The second frame goes next. This sequence goes on to the third frame, the fourth frame, and so on. With the proposed cache pruning, the context is examined frame by frame. The context distance for each frame is calculated based on equation 4.4. The context distance measures how far away the previous frames deviate from the current reference frame. The frame with the longest context distance is the candidate for deletion. In this scenario, frame 5 is picked as the best candidate. Based on the distance score frame 2 will go next. By examining frame 5, we see frame 5 corresponds to the mobile user going by Terry Bank. This frame is clearly out of the context by the time user goes by Publix at frame 7.

In scenario 3, the mobile user is roaming in the Dolphin mall. When it comes to cache prune, LRU deletes frames in the following order, 1, 2, 3, …, and 7. Context cache prune calculates the context distance for each frame. In Figure 5.14, frame 6 has the highest score, so it will be deleted from memory first. The next candidate will be frame 5. If we examine frame 6, we see the mobile user is at World Time. This frame is not close to the current user context.



```
Cache Prune
printVector(): Least Recently Used Cache Prune
frame 1
frame 2
frame 3
frame 4
frame 5
frame 6
frame 7
printVector(): Context Cache Prune
frame 1 context distance: 3.098571747217573
frame 2 context distance: 3.276340108799494
frame 3 context distance: 1.2070883634139755
frame 4 context distance: 1.251855607526343
frame 5 context distance: 3.466705218720252
frame 6 context distance: 2.4422976497653623
frame 7 context distance: 2.4224496898090564
```

Figure 5.15 Cache Prune Comparison for Scenario 2: Roaming off FIU campus



```
printVector(): Least Recently Used Cache Prune
frame 1
frame 2
frame 3
frame 4
frame 5
frame 6
frame 7
printVector(): Context Cache Prune
frame 1 context distance: 5.544991164235141
frame 2 context distance: 4.744652046837145
frame 3 context distance: 3.877311350116977
frame 4 context distance: 3.524769305011041
frame 5 context distance: 6.664382679579466
frame 6 context distance: 10.047563819874437
frame 7 context distance: 5.91702547206764
```

Figure 5.16 Cache Prune Comparison for Scenario 3: Shopping in the Dolphin Mall

CHAPTER VI

CONCLUSION

A major contribution of this dissertation is in proposing a context aware design methodology that allows a context aware application developer to focus more on the context-aware application itself rather dealing with low-level hardware which acquires the context data. Context awareness within this design concept defines the user context in a changing environment under different situations, and provides solutions for useful applications.

Context data acquisition in a cost-effective way using current technology is highly feasible, although the focus of this dissertation is placed on the search application with all its support algorithms. This would necessitate an infrastructure that builds on cost-effective technology such as Wi-Fi, Bluetooth, RFID, and ad hoc sensor networks. But, such a feat would require the participation of wireless carriers such as Verizon, ATT, and Sprint to name a few. Unfortunately, before we can reach a compromise between limited range or mobility and the much faster download speed or high bandwidth, the current trend remains more towards the deployment of the third generation network, which is broadband telecommunications and data communications.

The context aware architecture at the server side provides a platform to share the user context among context-aware applications. Context reusability shields application developers from low level context acquisition. The application design is much simplified.

The required architecture is envisioned to accommodate third-party service providers. Service provider could register their services and products with the designed architecture. The architecture provides user service and product based on the user current situation. A context search manager is developed to provide search function based on the current user situation. A user feedback mechanism augments this new search engine. The flowchart of the context-aware application demonstrates how the different software components at the server side interact to solve the user queries.

Virtual frame and reference frame are introduced to reflect the mobile user’s activities and surrounding environment. A proxy is added as a consequence to process user profiles and context frames. A new context cache scheme is provided to improve the cache hit for the data cache. The flowchart of the cache scheme shows how the proxy at client side monitors user context, calculates virtual/reference frames, and works with the cache scheme.

Furthermore, in order to address the issues of variability and instability in the notion of user context, the approach considered addressed the source of the difficulty in context awareness at the client side through the extraction of useful feature/context from user situations that are dynamic in nature. It is therefore viewed as important to design new algorithms at the client side to undertake the preliminary context analysis of the user situation. The client proxy should then relieve the networks (i.e., servers) of the computing burden.

To meet real-world needs required of a testing environment without the imposition of a heavy cost such an infrastructure will demand, a free Yahoo search API is thus integrated as a means to evaluate the effectiveness of the proposed design structure in a most realistic way. The test results show that the overall design is highly effective, providing new features and enriching the mobile user's experience through a broad scope of potential applications. The evaluation methods and many contemplated scenarios of the proposed architecture, search engine, proxy, and cache scheme have all proved successful. Yahoo search engine provides free API to programmers, which makes this type of simulation possible with real time search results. The simulation results are thus tightly dependent on the accuracy of the Yahoo search engine. However, the user feedback is implemented on the server side to further improve the search accuracy. The test results indicate that the design is robust in accomplishing the set goals. The context design, which includes various solutions at both server and client sides can, as demonstrated, be utilized to facilitate the mobile users in the future.

In retrospect, the merits of the proposed design structure can be drawn from the optimization steps that were undertaken at the different design levels, yielding the following accomplishments:

- Simplification on the hardware logic required of mobile devices

  Context-aware application at client side does not necessarily require a set of sensors. Context server is readily available to provide user context for such context-aware application. In addition, the proposed proxy at the client side is also capable of feeding extracted user context to context-aware applications. User context distribution is the key for such hardware simplification.

- Provision of context information with appropriate levels of interpretation and abstraction. Context modeling addresses the issue of data interpretation and abstraction. Context modeling can further represent data in appropriate format to satisfy the needs of context-aware application.
- Freeing application developers from low-level sensor details. With the combined solutions and support from both sever side and client side, context-aware application developers could focus on the feature design (GPS, location dependable services, and context-aware search). The user context can be requested from context server.
- Enhanced management and sharing of the user context among mobile devices. The deployment of the required context-aware infrastructure will certainly reduce the computing burden of the mobile devices. The other benefit is to share the user context among different applications.
- Added ability to configure dynamically new context-aware applications. As new applications are developed, installing and managing these applications on the mobile devices will be much easier, since context server can meet the context demand from these new applications.
- Provision of run time context-aware service registry and matching. The required context-aware infrastructure provides means for third-party vendors to register their products and services. These products and services are promoted to the targeted mobile users based on their context.

To reach of all these accomplishments, a hardware infrastructure was thus conceived with appropriate software developments as the different enumerated constraints related mainly to cost and time to deployment are kept in mind.

Finally, in order to facilitate replicating the research efforts of this dissertation and augmenting this broad research area, all the source code is provided in the Appendices, which include:

Appendix A. Client Side Implementation Code.

Appendix B. Client Side Search Implementation Module

Appendix C. Server Side Implementation Code

Appendix D. Code for Vendor Side Server Interface

LIST OF REFERENCES

[1] G. E. Moore, "Progress in Digital Integrated Electronics," in *Technical Digest of the Int'l Electron Devices Meeting*, 1975, pp. 11-13.

[2] A. C. Yamin, J. V. Barbosa, I. Augustin, L. C. da Silva, R. Real, C. Geyer, and G. Cavallheiro, "Towards merging context-aware, mobile and grid computing," *International Journal of High Performance Computing Applications,* vol. 17, pp. 191-203, 2003.

[3] T. Instruments, "OMAP^{TM} 3 family of multimedia applications processors," Texas Instruments.

[4] M. Weiser, "The Computer for the Twenty-First Century," *Scientific American,* vol. 265, pp. 94-104, 1991.

[5] A. Ranganathan and R. Campbell, "A Middleware for Context-Aware Agents in Ubiquitous Computing Environments," *Middleware* vol. 2672, pp. 143-161, 2003.

[6] J. Burrell, G. K. Gay, K. Kubo, and N. Farina, "Context-Aware Computing: A Test Case," in *the 4th international conference on Ubiquitous Computing*. vol. 2498 Göteborg, Sweden: Lecture Notes In Computer Science, 2002, pp. 647-653.

[7] P. E. Agre, "Changing places: Contexts of awareness in computing," *Human-Computer Interaction,* vol. 16, pp. 177-192, 2001.

[8] A. K. Dey and G. K. Abowd, "Understanding and using context," *Journal of Personal and Ubiquitous Computing,* vol. 5, pp. 4-7, 2001.

[9] G. Chen and D. Kotz, "A Survey of Context-Aware Mobile Computing Research," in *TR2000-381* Dartmouth College, 2000.

[10] K. Rehman, F. Stajano, and G. Coulouris, "An Architecture for Interactive Context-Aware Applications," *IEEE Pervasive Computing,* vol. 6,, pp. 73-80, 2007.

[11] D. Siewiorek, A. Smailagic, J. Furukawa, A. Krause, N. Moraveji, K. Reiger, J. Shaffer, and W. F. Lung, "SenSay: a context-aware mobile phone " in *Seventh IEEE International Symposium on Wearable Computers*, 2003, pp. 248-249.

[12] L. Capra, W. Emmerich, and C. Mascolo, "CARISMA: Context-aware reflective mlddleware system for mobile applications," 2003, pp. 929-945.

[13] N. Ryan, J. Pascoe, and D. Morse, "Enhanced Reality Fieldwork: the Context Aware Archaeological Assistant," in *the 25th Anniversary Computer Applications in Archaeology 1997*, 1998.

[14] W. N. Schilit, N. I. Adams, and R. Want, "Context-aware computing applications," in *IEEE Workshop on Mobile Computing Systems and Applications*, Santa Cruz, CA, USA, 1994, pp. 85-90.

[15] J. Wejchert, "The Disappearing Computer," in *1st Call for proposals,Future and Emerging Technologies*, 2000.

[16] P. Luff and C. Heath, "Mobility in Collaboration," in *the 1998 ACM conference on Computer supported cooperative work*, Seattle, Washington, 1998, pp. 305 - 314

[17] S. Tamminen, A. Oulasvirta, and K. Toiskallio, " Understanding mobile contexts," *Personal and Ubiquitous Computing,* pp. 135-143, 2004.

[18] W. Y. Lum and F. C. M. Lau, "User-centric content negotiation for effective adaptation service in mobile computing," *Software Engineering, IEEE Transactions on,* vol. 29, pp. 1100 - 1111, 2003

[19] T. Gu, H. K. Pung, and D. Q. Zhang, "A service-oriented middleware for building context-aware services," *Journal of Network and Computer Applications,* vol. 28, pp. 1-18, Jan 2005.

[20] G. Cabri, L. Leonardi, and F. Zambonelli, "Engineering mobile agent applications via context-dependent coordination," *Ieee Transactions on Software Engineering,* vol. 28, pp. 1039-1055, Nov 2002.

[21] H. Maass, "Location-aware mobile applications based on directory services," *Mobile Networks and Applications,* vol. 3, pp. 157 - 173 1998.

[22] E. E. P. 26900, "Technology for enabling Awareness (TEA)," 1998.

[23] H. Keranen, T. Rantakokko, and J. Mantyarvi, "Sharing and presenting multimedia and context information within online communities using mobile terminals," in *ICME 2003 International Conference on Multimedia and Expo*, 2003, pp. 641-644.

[24] C. Julien and G. C. Roman, "EgoSpaces: Facilitating rapid development of context-aware mobile applications," *Ieee Transactions on Software Engineering,* vol. 32, pp. 281-298, May 2006.

[25] B. N. Schilit, A. LaMarca, D. McDonald, J. Tabert, E. Cadag, G. Borriello, and W. G. Griswold, "Bootstrapping the Location-enhanced World Wide Web," in *Location-Aware Computing Workshop (UbiComp 2003)*, 2003.

[26] J. E. Bardram, "Applications of Context-Aware Computing in Hospital Work - Examples and Design Principles," in *the 2004 ACM symposium on Applied computing* Nicosia, Cyprus, 2004 pp. 1574 - 1579

[27] D. Salber, A. K. Dey, and G. D. Abowd, "The context toolkit: aiding the development of context-enabled applications," in *the SIGCHI conference on Human factors in computing systems: the CHI is the limit* Pittsburgh, Pennsylvania, United States, 1999 pp. 434 - 441

[28] P. Fahy and S. Clarke, "Context-aware middleware for mobile multimedia applications," in *the 3rd international conference on Mobile and ubiquitous multimedia table of contents*. vol. 83 College Park, Maryland, 2004, pp. 213 - 220

[29] H. Chen, "An Intelligent Broker Architecture for Pervasive Context-Aware Systems." vol. PhD thesis Baltimore County: University of Maryland, 2004.

[30] A. Bhattacharya, R. A, and S. K. Das, "Towards a Novel Architecture to Support Universal Location Awareness," *GI Jahrestagung,* vol. 1, pp. 182-189, 2001.

[31] H. W. Gellersen, A. Schmidt, and M. Beigl, "Multi-sensor context-awareness in mobile devices and smart artifacts," 2002, pp. 341-351.

[32] N. Bulusu, J. Heidemann, and D. Estrin, " GPS-less low cost outdoor localization for very small devices " *IEEE Personal Communications Magazine* vol. 7, pp. 28-34, 2000.

[33] Y. Chen and Q. Zhao, "Wireless Sensor Networks " *IEEE Commun. Let,* vol. 9, pp. 976-978, 2004.

[34] P. Korpipaa and J. Mantyjarvi, "An ontology for mobile device sensor-based context awareness," *Modeling and Using Context,* pp. 451-458, 2003.

[35] Y. Cui and V. Roto, "How People Use the Web on Mobile Devices," in *the 17th international conference on World Wide Web* Beijing, China 2008, pp. 905-914

[36] P. D. Michailidis and K. G. Margaritis, "On-line String Matching Algorithms: Survey and Experimental Results," *International journal of computer mathematics* vol. 76, pp. 411-434 2001.

[37] M. S. Lew, N. Sebe, C. Djeraba, and R. Jain, "Content-based multimedia information retrieval: State of the art and challenges," *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMCCAP),* vol. 2, pp. 1 - 19 2006.

[38] M. Kamvar and S. Baluja, " Deciphering Trends in Mobile Search," *Computer,* vol. 40 pp. 58-62, 2007.

[39] M. Kellar, C. Watters, and M. Shepherd, "A Goal-based Classification of Web Information Tasks," in *the Annual Meeting of the American Society for Information Science and Technology* Austin, Texas, 2006.

[40] O. Software, "State of the Mobile Web Report: First Quarter," 2008.

[41] A. Soffer, Y. Maarek, and B. W. Chang, "WWW2002 Workshop on Mobile Search," Honolulu, Hawaii2002.

[42] B. Meunier, "Characteristics of the Top 100 Mobile Search Queries at AT&T”," 2008.

[43] B. J. Jansen, D. L. Booth, and A. Spink, "Determining the User Intent of Web Search Engine Queries," in *16th international conference on World Wide Web*, Banff,Alberta, Canada, 2007, pp. 1149 - 1150

[44] S. Cronen-Townsend and W. B. Croft, "Quantifying Query Ambiguity," in *the second international conference on Human Language Technology Research*, San Diego,California, 2002, pp. 104 - 109.

[45] R. Song, Z. X. Luo, J. R. Wen, and H. W. Hon, "Identifying Ambiguous Queries in Web Search," in *the 16th international conference on World Wide Web* Banff, Alberta, Canada,, 2007, pp. 1169 - 1170.

[46] X. H. Shen, B. Tan, and C. X. Zhai, "Implicit User Modeling for Personalized Search," in *the 14th ACM international conference on Information and knowledge management*, Bremen, Germany, 2005, pp. 824 - 831.

[47] F. Qiu and J. H. Cho, "Automatic Identification of User Interest for Personalized Search," in *the 15th international conference on World Wide Web*, Edinburgh, UK, 2006, pp. 727 - 736

[48] P. A. Chirita, C. S. Firan, and W.Nejdl, "Summarizing Local Context to Personalize Global Web Search," in *the 15th ACM international conference on Information and knowledge management*, Arlington, Virginia, 2006, pp. 287 - 296

[49] Z. C. Dou, R. H. Song, and J. R. Wen, "A Large-scale Evaluation and Analysis of Personalized Search Strategies," in *the 16th international conference on World Wide Web*, Banff, Alberta, Canada 2007, pp. 581 - 590.

[50] J. Teevan, S. T. Dumais, and D. J. Liebling, "To Personalize or Not to Personalize: Modeling Queries with Variation in User Intent," in *the 31st annual international ACM SIGIR conference on Research and development in information retrieval* Singapore, Singapore 2008.

[51] J. Liu and L. Birnbaum, "What do they think? Aggregating Local Views about News Events and Topics," in *the 17th international conference on World Wide Web*, Beijing, China, 2008, pp. 1021-1022

[52] S. Vadrevu, Y. Zhang, B. Tseng, G. Sun, and X. Li, "Identifying Regional Sensitive Queries in Web Search," in *the 17th international conference on World Wide Web*, Beijing, China, 2008, pp. 1185-1186

[53] I. Chlamtaca and J. Redi, "Mobile Computing: Challenges and Potential " in *Encyclopedia of Computer Science*, 4th ed: International Thomson Publishing, 1998.

[54] M. Adjouadi and M. Ayala, "Introducing Neural Studio: An Artificial Neural Networks simulation for Educational Purpose," *Computers in Education Journal,* vol. 14, pp. 33-40, 2004.

[55] B. Clarkson and A. Pentland, " Unsupervised Clustering of Ambulatory Audio and Video " in *IEEE International Conference on Acoustics, Speech, and Signal Processing,*, Phoenix, AZ, USA, 1999, pp. 3037-3040.

[56] J. Himberg, J. A. Flanagan, and J. Mäntyjärvi, " Towards Context Awareness Using Symbol Clustering Map," in *Workshop on Self-Organising Maps WSOM2003*, Hibikino, Kitakyushu, Japan, 2003, pp. 249--254.

[57] T. Kohonen, *Self-Organizing Map*, Third Extended ed. Berlin, Heidelberg: Springer, 2001.

[58] A. Schmidt, K. A. Aidoo, A. Takaluoma, U. Tuomela, K. V. Laerhoven, and W. V. Velde, "Advanced Interaction in Context," in *First International Symposium on Handheld and Ubiquitous Computing*, 1999, pp. 89-101.

[59] E. J. O'Neil, P. E. O'Neill, and G. Weikum, "The LRU-K Page Replacement Algorithm For Database Disk Buffering," in *the 1993 ACM SIGMOD international conference on Management of data* Washington, D.C., United States 1993, pp. 297 - 306.

[60] J. T. Robinson and M. V. Devarakonda, " Data Cache Management Using Frequency-Based Replacement " in *the 1990 ACM SIGMETRICS conference on Measurement and modeling of computer systems* Boulder, Colorado, United States, 1990, pp. 134 - 142.

APPENDICES

Appendix A. Client Side Implementation Code.

```
/*
 * Author :
 *       Feng Gui
 * Description:
 * Proxy implements the functions proposed in the dissertation.
 * It has the user interface to trigger the ANN to calculate
 * virtual frame and reference frame. It also trigger the cache
 * prune function.
 */

import java.util.*;
import java.io.*;
import java.awt.*;
import java.sql.*;
import javax.swing.*;
import javax.swing.border.*;

public class Proxy
{
  // user interface components.
  static final int FRAME_WIDTH=800, FRAME_HEIGHT=500,
BORDER_OFFSET=50;
  static private JFrame jframe;
  static private JEditorPane jep;
  static private JScrollPane jsp;
  static private JPanel bottom_panel;
  static private JButton close_button, cache_button;
  static private String file_name = "data3.txt";
  static private MySQL mysql;
  static private Cache cache;

  // global total_iteration refers to the total iteration number
  // variable total_entities refers to the total entity count for all iterations.
  // Both are not zero based.
  static private int total_iterations, total_entities = 0;
  static private Vector<MyData> context_data, virtual_frame, reference_frame;

  public static void main(String[] args) throws IOException
  {
    mysql = new MySQL();
    context_data = new Vector<MyData>();
    createFrame();
    readFile(file_name);
    printVector("context data", context_data);
    double mu = 0.005;
    Ann ann = new Ann(total_iterations, total_entities, context_data, mu);
    virtual_frame = ann.calculateVirtualFrame();
    printVector("virtual frame", virtual_frame);
    reference_frame = ann.calculateReferenceFrame();
    printVector("reference frame", reference_frame);
    ann.uploadReferenceFrames();
  }

  private static void createFrame()
  {
    // Uncomment the following line will set the look and feel in java style.
    //JFrame.setDefaultLookAndFeelDecorated(true);

    jframe = new JFrame("Proxy");
    jframe.setSize(FRAME_WIDTH, FRAME_HEIGHT);
    jframe.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    jframe.getContentPane().setLayout(new BorderLayout());

    // The icons folder must be placed under project proxy root folder.
    Image icon = Toolkit.getDefaultToolkit().getImage("icons/RSS_001.png");
    jframe.setIconImage(icon);

    // Create the text area.
    jep = new JEditorPane();
    jep.setEditable(false);
    jsp = new JScrollPane(jep);
    jsp.setPreferredSize(new Dimension(jframe.WIDTH-BORDER_OFFSET,
jframe.HEIGHT-BORDER_OFFSET));
    jframe.getContentPane().add(jsp, BorderLayout.CENTER);
    jep.setText("initialize ... ");

    // Create button panel at bottom.
    bottom_panel = new JPanel();
    Dimension buttonSize = new Dimension(80,20);
    cache_button = new JButton("Cache");
    cache_button.setPreferredSize(buttonSize);
    cache_button.addMouseListener(new java.awt.event.MouseAdapter()
                    {
                      public void mouseClicked(java.awt.event.MouseEvent evt)
```

```
                            {
                              cachePrune();
                            }
                          });
    bottom_panel.setBorder(new
CompoundBorder(BorderFactory.createLineBorder(Color.LIGHT_GRAY, 2),
BorderFactory.createBevelBorder(BevelBorder.LOWERED)));
    bottom_panel.add(cache_button);

    close_button = new JButton("Close");
    close_button.setPreferredSize(buttonSize);
    close_button.addMouseListener(new java.awt.event.MouseAdapter()
                          {
                            public void mouseClicked(java.awt.event.MouseEvent evt)
                            {
                              System.exit(1);
                            }
                          });
    bottom_panel.add(close_button);
    jframe.getContentPane().add(bottom_panel, BorderLayout.SOUTH);

    centerFrame(jframe);

    jframe.setVisible(true);
  }

  private static void cachePrune()
  {
    print("\nCache Prune");

    cache = new Cache(reference_frame, virtual_frame);
    double[] LRUFrames = cache.LRUCachePrune();
    printVector("LRU", LRUFrames);

    double[] context_distances = cache.contextCachePrune();
    printVector("context", context_distances);
  }

  private static void centerFrame(JFrame jframe)
  {
    // Center the main frame only if the frame size is smaller than the window screen.
    Toolkit windowScreen = jframe.getToolkit();
    Dimension windowSize = windowScreen.getScreenSize();
    if (FRAME_WIDTH < windowSize.width && FRAME_HEIGHT <
windowSize.height)
```

```
      jframe.setBounds((windowSize.width-FRAME_WIDTH)/2, (windowSize.height-
FRAME_HEIGHT)/2, FRAME_WIDTH, FRAME_HEIGHT);
    else
      jframe.setBounds(0,0,windowSize.width,windowSize.height);
  }

  private static void readFile(String file_name)
  {
    total_iterations = 0;
    PreparedStatement ps;
    Connection connection;

    try
    {
      connection = mysql.getConnection();
      ps = connection.prepareStatement("truncate location");
      ps.executeQuery();

      BufferedReader br = new BufferedReader(new FileReader(file_name));
      String line;

      // clear jep first
      jep.setText("Data file: "+file_name+"\n");
      while ((line = br.readLine()) != null)
      {
        print("readFile()"+line);
        loadData(line, total_iterations++);
      }
      br.close();
      connection.close();
    }
    catch (IOException e)
    {
      jep.setText(e.getMessage());
    }
    catch(ClassNotFoundException ex)
    {
      JOptionPane.showMessageDialog(null, "Mysql driver class not found.", "Alert",
JOptionPane.ERROR_MESSAGE);
    }
    catch(SQLException ex)
    {
      JOptionPane.showMessageDialog(null, ex.getMessage(), "Alert",
JOptionPane.ERROR_MESSAGE);
      //e.printStackTrace();
    }
  }

  // Each line basically represent one iteration data.
  private static void loadData(String line, int iteration)
  {
    int e, TYPE=0, SOURCE=1, VALUE=1;
    float x=0, y=0;
    String location = "";
    String[] context_entities, context_entity, types, sources;
    MyData data;
    PreparedStatement ps;
    Connection connection;

    try
    {
      connection = mysql.getConnection();


      if (line != null && line.length()>0)
      {
        context_entities = line.split(";");

        if (total_entities < context_entities.length)
          total_entities = context_entities.length;

        for (e=0; e<context_entities.length; e++)
        {
          x=0;
          y=0;
          context_entity = context_entities[e].split(",");
          //print("loadData(): "+entity[TYPE]+" "+entity[SOURCE]);

          types = context_entity[TYPE].split("=");
          sources = context_entity[SOURCE].split("=");

          data = new MyData(types[TYPE], Double.valueOf(types[VALUE]),
Integer.valueOf(sources[VALUE]), iteration, iteration, e);
          print("loadData(): create MyData "+types[TYPE]+"
"+Double.valueOf(types[VALUE])+" "+Integer.valueOf(sources[VALUE])+"
"+iteration+" "+iteration+" "+e);
          context_data.add(data);

          // load the location table for this user.
          if (types[TYPE].toLowerCase().equals("x"))
```

```
            x = Float.valueOf(types[VALUE]);
          else if (types[TYPE].toLowerCase().equals("y"))
            y = Float.valueOf(types[VALUE]);
          else if (types[TYPE].equals("location") || types[TYPE].equals("store"))
            location = types[VALUE];
        }

        ps = connection.prepareStatement("select * from location_class where  class =
?");
        ps.setInt(1, Integer.valueOf(location));
        ResultSet rs = ps.executeQuery();

        if (rs.next())
          location = rs.getString("location");

        ps = connection.prepareStatement("insert into location values (?, ?, ?, ?, ?)");
        ps.setInt(1, iteration);
        ps.setFloat(2, Float.valueOf(x));
        ps.setFloat(3, Float.valueOf(y));
        ps.setString(4, location);
        ps.setTimestamp(5, new Timestamp(System.currentTimeMillis()));
        ps.executeUpdate();
      }
      connection.close();
    }
    catch(ClassNotFoundException ex)
    {
      JOptionPane.showMessageDialog(null, "Mysql driver class not found.", "Alert",
JOptionPane.ERROR_MESSAGE);
    }
    catch(SQLException ex)
    {
      JOptionPane.showMessageDialog(null, ex.getMessage(), "Alert",
JOptionPane.ERROR_MESSAGE);
      //e.printStackTrace();
    }
  }

  private static void print(String line)
  {
    jep.setText(jep.getText()+line+"\n");
  }

  private static void printVector(String type, double[] v)
  {
    int i;
    if (v.length>0)
    {
      if (type.equals("context"))
        print("\nprintVector(): Context Cache Prune");
      else
        print("\nprintVector(): Least Recently Used Cache Prune");

      for (i=0; i<v.length; i++)
      {
        if (type.equals("context"))
          print("\nframe "+(i+1)+" context distance: "+v[i]);
        else
          print("\nframe "+(i+1));
      }
    }
  }

  private static void printVector(String type, Vector<MyData> v)
  {
    int i, iteration = -1;
    String line = "";

    if (!v.isEmpty())
    {
      print("\nprintVector(): "+type);
      for (i=0; i<v.size(); i++)
      {
        if (iteration != v.get(i).getIteration())
        {
          print(line);
          iteration = v.get(i).getIteration();
          line = "printVector(): iteration "+iteration+":";
        }
        line += " (index="+v.get(i).getIndex()+",
"+v.get(i).getType()+"="+v.get(i).getValue()+") ";
      }
      print(line);
    }

  }
}

/*
 * This routine implements the cache algorithm proposed in the dissertation.
 */
import java.util.*;
import javax.swing.*;

public class Cache
{
  private Vector<MyData> last_reference_frame;
  private Vector<MyData> virtual_frames;

  public Cache(Vector<MyData> reference_frames, Vector<MyData> virtual_frames)
  {
    last_reference_frame = extractLastReferenceFrame(reference_frames);
    this.virtual_frames = virtual_frames;
    if (last_reference_frame != null && !last_reference_frame.isEmpty())
      printVector(last_reference_frame);
  }

  public double[] LRUCachePrune()
  {
    int i, iteration = -1;
    double[] frames = new double[0];

    if (!virtual_frames.isEmpty())
    {
      frames = new double[virtual_frames.get(virtual_frames.size()-1).getIteration()+1];
      for (i=0; i<virtual_frames.size(); i++)
      {
        if (iteration != virtual_frames.get(i).getIteration())
          iteration = virtual_frames.get(i).getIteration();
        frames[iteration] = iteration;
      }
    }

    return frames;
  }

  public double[] contextCachePrune()
  {
    int i, index = 0, iteration = -1;
    double[] distances = new double[0];

    if (!virtual_frames.isEmpty())
    {
      distances = new double[virtual_frames.get(virtual_frames.size()-
1).getIteration()+1];
```

```
      for (i=0; i<virtual_frames.size(); i++)
      {
        if (iteration != virtual_frames.get(i).getIteration())
        {
          index = 0;
          iteration = virtual_frames.get(i).getIteration();
        }

        distances[iteration] += Math.abs( last_reference_frame.get(index).getValue() -
virtual_frames.get(i).getValue() );
        index++;
      }
    }

    return distances;
  }

  private Vector<MyData> extractLastReferenceFrame(Vector<MyData>
reference_frames)
  {
    MyData data;
    Vector<MyData> reference_frame = new Vector<MyData>();
    int index, last_index, last_iteration;

    if (reference_frames == null || reference_frames.isEmpty())
      JOptionPane.showMessageDialog(null, "Reference frames not set", "Error",
JOptionPane.ERROR_MESSAGE);
    else
    {
      last_index = reference_frames.size()-1;
      last_iteration = reference_frames.get(last_index).getIteration();

      System.out.println("\nCache.java->extractLastReferenceFrame() last_iteration =
"+last_iteration);
      for (index=0; index<reference_frames.size(); index++)
      {
        if (reference_frames.get(index).getIteration() < last_iteration)
          continue;
        data = new MyData(reference_frames.get(index).getType(),
reference_frames.get(index).getValue(), reference_frames.get(index).getSource(),
reference_frames.get(index).getFrequency(), reference_frames.get(index).getIteration(),
reference_frames.get(index).getIndex());
        reference_frame.add(data);
      }
    }
    return reference_frame;
  }

  private static void printVector(Vector<MyData> v)
  {
    int i, iteration = -1;
    String line = "";

    if (!v.isEmpty())
    {
      for (i=0; i<v.size(); i++)
      {
        if (iteration != v.get(i).getIteration())
        {
          System.out.println(line);
          iteration = v.get(i).getIteration();
          line = "Cache.java->printVector(): iteration "+iteration+":";
        }
        line += " (index="+v.get(i).getIndex()+",
"+v.get(i).getType()+"="+v.get(i).getValue()+") ";
      }
      System.out.println(line);
    }
  }
}
/*
 * ANN stands for artificial neural network
 * Ann implements the algorithm I proposed in the dissertation.
 * It calculates the virtual frame and reference frame.
 */


import java.sql.*;
import java.util.*;
import javax.swing.*;

public class Ann
{
  // last_iteration and last_index are not zero based
  private int last_iteration, last_index;
  private Vector<MyData> context_data, virtual_frame, reference_frame;
  private double mu;
  private MySQL mysql;
  public Ann(int total_iterations, int total_entities, Vector<MyData> context_data,
double mu)
  {
    virtual_frame = new Vector<MyData>();
    reference_frame = new Vector<MyData>();
    last_iteration = total_iterations;
    last_index = total_entities;
    this.context_data = context_data;
    this.mu = mu;
    mysql = new MySQL();
  }

  /**
   * In essence, the reference frame is the moving average of the virtual frame.
   * @return
   */
  public Vector<MyData> calculateReferenceFrame()
  {
    MyData data;
    int iteration=0, index;
    double[] sums = new double[last_index], averages = new double[last_index];

    if (!virtual_frame.isEmpty())
    {
      for (index=0; index<virtual_frame.size(); index++)
      {
        if (virtual_frame.get(index).getIndex() == 0)
          iteration = virtual_frame.get(index).getIteration();

        sums[virtual_frame.get(index).getIndex()] +=
virtual_frame.get(index).getValue();
        if (iteration>0)
        {
          averages[virtual_frame.get(index).getIndex()] =
sums[virtual_frame.get(index).getIndex()]/(iteration+1);
          //MyData(String type, double value, int source, int frequency, int iteration, int
index)
          data = new MyData(virtual_frame.get(index).getType(),
averages[virtual_frame.get(index).getIndex()], virtual_frame.get(index).getSource(),
virtual_frame.get(index).getFrequency(), virtual_frame.get(index).getIteration()-1,
virtual_frame.get(index).getIndex());
          reference_frame.add(data);
        }

      }
```

```
        System.out.print("Ann.java->calculateReferenceFrame():");
        for (index=0; index<last_index; index++)
        {
          if (index == 0)
            System.out.print(sums[index]);
          else
            System.out.print(", "+sums[index]);
        }
        System.out.println();
      }
      return reference_frame;
    }

    public Vector<MyData> calculateVirtualFrame()
    {
      int iteration, index;

      MyData data;

      double weight_change = 0, previous_weight = 0, weight = 0;

      // Note: iteration = 0, all context weights are initialzed.
      for (iteration=0; iteration<last_iteration; iteration++)
      {
        System.out.println("Ann.java->calculateVirtualFrame(): iteration="+iteration);
        // initialize first iteration
        if (iteration==0)
        {
          for (index=0; index<last_index; index++)
          {
            //MyData(String type, double value, int source, int frequency, int iteration, int
index)
            data = new MyData(context_data.get(index).getType(), 1,
context_data.get(index).getSource(), 1, 0, index);
            virtual_frame.add(data);
            //System.out.print("(index:"+virtual_frame.get(index).getIndex()+",
weight:"+virtual_frame.get(index).getValue()+") ");
          }
        }
        // the weights in higher iterations are derived from previous iteration.
        else
        {
          for (index=0; index<last_index; index++)
          {
            previous_weight = getWeight("virtual", iteration-1, index);
            weight_change = getDifference(context_data, iteration, index,
mu)*previous_weight;
            weight = previous_weight + weight_change;
            data = new MyData(context_data.get(index).getType(), weight,
context_data.get(index).getSource(), context_data.get(index).getFrequency()+1, iteration,
index);
            virtual_frame.add(data);
            System.out.println("Ann.java->calculateVirtualFrame(): index="+index+",
previous_weight="+previous_weight+", weight_change="+weight_change+",
weight="+weight);
          }
        }
      }

      printVector(virtual_frame);

      return virtual_frame;
    }

    /**
     *
     * @param data: one dimentional vector which stores all the context entity data
     * @param iteration
     * @param index: entity in the current iteration
     * @return: |(change in context entity) - u*(change in frequency)|
     */
    private double getDifference(Vector<MyData> context_data, int iteration, int index,
double mu)
    {
      int d, FREQUENCY_THRESHOLD = 2;
      double previous_context = -1, current_context = -1, sum = 0, change = 0,
            previous_frequency = -1, current_frequency = -1, context_change = -1,
frequency_change = -1;

      if (!context_data.isEmpty())
      {
        for (d=0; d<context_data.size(); d++)
        {
          // In case the context difference is zero.
          if (context_data.get(d).getIndex() == index)
            sum += context_data.get(d).getValue();

          if (context_data.get(d).getIteration() == iteration - 1 &&
context_data.get(d).getIndex() == index)
```

```
        {
          previous_context = context_data.get(d).getValue();
          previous_frequency = context_data.get(d).getFrequency();
          //System.out.println("Ann.java->getDifference(): Iteration="+iteration+",
index="+index+", previous_context="+previous_context);
        }

        if (context_data.get(d).getIteration() == iteration &&
context_data.get(d).getIndex() == index)
        {
          current_context = context_data.get(d).getValue();
          current_frequency = context_data.get(d).getFrequency();
          //System.out.println("Ann.java->getDifference(): Iteration="+iteration+",
index="+index+", current_context="+current_context);
          // once get the data, break off.
          break;
        }
      }

      if (previous_context != 0)
        context_change = Math.abs((current_context -
previous_context)/previous_context);
      else
        context_change = Math.abs(current_context - previous_context);

      // if no change in context entity, then take average up to current iteration.
      if (context_change == 0)
        context_change = Math.abs(current_context - sum / (iteration+1))*(-1);

      frequency_change = current_frequency - previous_frequency;
      if (frequency_change == 0 && frequency_change >
FREQUENCY_THRESHOLD)
        frequency_change = current_frequency - 1;

      change = context_change + mu*frequency_change;
      System.out.println("Ann.java->getDifference(): index="+index+",
pre_context="+previous_context+", current_context="+current_context+",
sum="+sum+", context_change="+context_change+",
frequency_change="+frequency_change+", change="+change);
    }
    return change;
  }

  private double getWeight(String type, int iteration, int index)
  {
    Vector<MyData> v = new Vector<MyData>();

    if (type.equals("virtual"))
      v = virtual_frame;
    else if (type.equals("refrence"))
      v = reference_frame;

    int w;

    if (!v.isEmpty())
    {
      for (w=0; w<v.size(); w++)
      {
        if (v.get(w).getIteration() == iteration && v.get(w).getIndex() == index)
          return v.get(w).getValue();
      }
    }

    return 0;
  }

  private static void printVector(Vector<MyData> v)
  {
    int i, iteration = -1;
    String line = "";

    if (!v.isEmpty())
    {
      for (i=0; i<v.size(); i++)
      {
        if (iteration != v.get(i).getIteration())
        {
          System.out.println(line);
          iteration = v.get(i).getIteration();
          line = "Ann.java->printVector(): iteration "+iteration+":";
        }
        line += " (index="+v.get(i).getIndex()+",
"+v.get(i).getType()+"="+v.get(i).getValue()+") ";
      }
      System.out.println(line);
    }
  }

  public void uploadReferenceFrames()
  {
```

```
    int w;
    PreparedStatement ps;
    Statement statement;
    Connection connection;

    try
    {
      // Clear reference table first.
      statement = mysql.getConnectionStatement();
      statement.executeUpdate("truncate table reference");
      statement.close();

      connection = mysql.getConnection();
      ps = connection.prepareStatement("insert into reference values (?, ?, ?, ?, ?, ?, ?)");

      for (w=0; w<reference_frame.size(); w++)
      {
        ps.setString(1, reference_frame.get(w).getType());
        ps.setString(2, String.valueOf(reference_frame.get(w).getValue()));
        ps.setInt(3, reference_frame.get(w).getSource());
        ps.setInt(4, reference_frame.get(w).getFrequency());
        ps.setInt(5, reference_frame.get(w).getIteration());
        ps.setInt(6, reference_frame.get(w).getIndex());
        //ps.setDate(7, new java.sql.Date(System.currentTimeMillis()));
        ps.setTimestamp(7, new Timestamp(System.currentTimeMillis()));

        ps.executeUpdate();
      }
      connection.close();
    }
    catch(ClassNotFoundException e)
    {
      JOptionPane.showMessageDialog(null, "Mysql driver class not found.", "Alert",
JOptionPane.ERROR_MESSAGE);
    }
    catch(SQLException e)
    {
      JOptionPane.showMessageDialog(null, e.getMessage(), "Alert",
JOptionPane.ERROR_MESSAGE);
      //e.printStackTrace();
    }
  }
}
/*
 * This is the foundation class for other program to store the
 * context entity.
 */

public class MyData
{
  private String type;
  private double value;
  private int source;
  private int frequency;
  private int iteration;
  private int index;

  /** Creates a new instance of MyData */
  public MyData(String type, double value, int source, int frequency, int iteration, int
index)
  {
    this.type = type;
    this.value = value;
    // source value:
    // 1: system generated input data
    // 2: user input data
    this.source = source;
    this.frequency = frequency;
    this.iteration = iteration;
    this.index = index;
  }

  // Copy constructor
  public MyData(MyData cd)
  {
    this.type = cd.type;
    this.value = cd.value;
    this.frequency = cd.frequency;
    this.iteration = cd.iteration;
    this.index = cd.index;
  }

  public String getType()
  {
    return type;
  }

  public int getSource()
  {
    return source;
  }

  public double getValue()
  {
    return value;
  }

  public void updateIndex(int index)
  {
    this.index = index;
  }

  public void updateValue(double value)
  {
    this.value = value;
  }

  public int getFrequency()
  {
    return frequency;
  }

  public int getIteration()
  {
    return iteration;
  }
  public int getIndex()
  {
    return index;
  }
}
```

```
/**
 *
 * Description:
 *
 * This program is the main interface program that shows on the mobile
 * device.
 * This program lists functions/features available to the user.
 * It shows different "screens" to the user based on the user election.
 */

import javax.microedition.midlet.MIDlet;
import javax.microedition.midlet.MIDletStateChangeException;
import javax.microedition.lcdui.*;


// To review the J2ME documents that come with the compiler
// Start -> All Programs -> Sun Java(TM) Wireless Toolkit 2.5.2 for CLDC ->
Documentation.

// Create a J2ME project under Eclipse SDK 3.3.2
// File -> New -> Project... -> J2ME -> J2ME Midlet Suite

// window "New J2ME Project" -> Project name: -> Next
// Under "Device", "Group:" select "Sun Java(TM) Wireless Toolkit 2.5.2 for CLDC
//       "Device:" select "MediaControlSkin" or other skins provided by the toolkit.
// Click "Next" if I want to import additional libraries such as kxml2-2.3.0.jar

// Source files are saved under "C:\Documents and
Settings\feng\workspace\guiClient\src",
// if Eclipse SDK is used.

public class guiClient extends MIDlet implements CommandListener
{
  //The display for this MIDlet
  private Display display;
  private Form form;

  private Proxy proxy;
  private final int ERROR=0, START_SCREEN=1, PORT_SCREEN=2,
DEFAULT_PORT=8888;
  private int port = 0;
  private TextField tf;
  private Command exit_command = new Command("Exit", Command.EXIT, 1);
  private Command start_command = new Command("Start", Command.ITEM, 1);
  private Command back_command = new Command("Back", Command.BACK, 0);
  private Command connect_command = new Command("Connect", Command.ITEM,
1);

  public guiClient()
  {
     // TODO Auto-generated constructor stub
            System.out.println("guiClient.java: guiClient() called");
            display = Display.getDisplay(this);
            show_form(START_SCREEN);
  }

  public void show_client()
  {
           display.setCurrent(form);
           show_form(PORT_SCREEN);
  }

  //protected void destroyApp(boolean arg0) throws MIDletStateChangeException
  protected void destroyApp(boolean arg0)
  {
    // TODO Auto-generated method stub
    display.setCurrent((Displayable)null);
  }

  protected void pauseApp()
  {
    // TODO Auto-generated method stub
  }

  protected void startApp() throws MIDletStateChangeException
  {
    // TODO Auto-generated method stub
    display.setCurrent(form);
  }

  public void commandAction(Command c, Displayable d)
  {
    if (c == exit_command)
    {
           System.out.println("guiClient.java: commandAction() exit");
           destroyApp(true);
           notifyDestroyed();
    }
    else if (c == start_command )
    {
            show_form(PORT_SCREEN);
        }
        else if (c == back_command)
        {
            show_form(START_SCREEN);
        }
        else if (c == connect_command )
        {
            port = Integer.parseInt(tf.getString());
            proxy = new Proxy(this);
            proxy.start();
        }
    }

    public void show_form(int type)
    {
            form = new Form("form");
            if (type == START_SCREEN)
            {
                    form.setTitle("Client Proxy");
                    form.addCommand(exit_command);
                    form.addCommand(start_command);
                    System.out.println("guiClient.java: show_form(START_SCREEN)
called.");
            }
            else if (type == PORT_SCREEN)
            {
                    form.setTitle("Set Connection");
                    tf = new TextField("Port number:", String.valueOf(DEFAULT_PORT),
                                        6, TextField.NUMERIC);
                    form.append(tf);
                    form.addCommand(back_command);
                    form.addCommand(connect_command);
                    System.out.println("guiClient.java: show_form(PORT_SCREEN)
called.");
            }
            form.setCommandListener(this);
            display.setCurrent(form);
    }

    public int get_port()
    {
            return port;
    }
}
```

Appendix B. Client Side Search Implementation Module

```
/*
*
* author: Feng Gui
*
* Description:
*
* This file is the main interface program that shows on the mobile
* device.
* This program lists functions/features available to the user.
* It shows different "screens" to the user based on the user selection.
*/
import javax.microedition.midlet.MIDlet;
import javax.microedition.midlet.MIDletStateChangeException;
import javax.microedition.lcdui.*;


// To review the J2ME documents that come with the compiler
// Start -> All Programs -> Sun Java(TM) Wireless Toolkit 2.5.2 for CLDC ->
Documentation.

// Create a J2ME project under Eclipse SDK 3.3.2
// File -> New -> Project... -> J2ME -> J2ME Midlet Suite

// window "New J2ME Project" -> Project name: -> Next
// Under "Device", "Group:" select "Sun Java(TM) Wireless Toolkit 2.5.2 for CLDC
//      "Device:" select "MediaControlSkin" or other skins provided by the toolkit.
// Click "Next" if I want to import additional libraries such as kxml2-2.3.0.jar

// Source files are saved under "C:\Documents and
Settings\feng\workspace\guiClient\src",
// if Eclipse SDK is used.

public class guiClient extends MIDlet implements CommandListener
{
  //The display for this MIDlet
  private Display display;
  private Form form;

  private Proxy proxy;
  private final int ERROR=0, START_SCREEN=1, PORT_SCREEN=2,
DEFAULT_PORT=8888;
  private int port = 0;
  private TextField tf;
  private Command exit_command = new Command("Exit", Command.EXIT, 1);
  private Command start_command = new Command("Start", Command.ITEM, 1);
  private Command back_command = new Command("Back", Command.BACK, 0);
  private Command connect_command = new Command("Connect", Command.ITEM,
1);

  public guiClient()
  {
    // TODO Auto-generated constructor stub
            System.out.println("guiClient.java: guiClient() called");
            display = Display.getDisplay(this);
            show_form(START_SCREEN);
  }

  public void show_client()
  {
            display.setCurrent(form);
            show_form(PORT_SCREEN);
  }

  //protected void destroyApp(boolean arg0) throws MIDletStateChangeException
  protected void destroyApp(boolean arg0)
  {
    // TODO Auto-generated method stub
    display.setCurrent((Displayable)null);
  }

  protected void pauseApp()
  {
    // TODO Auto-generated method stub
  }

  protected void startApp() throws MIDletStateChangeException
  {
    // TODO Auto-generated method stub
    display.setCurrent(form);
  }

  public void commandAction(Command c, Displayable d)
  {
    if (c == exit_command)
    {
          System.out.println("guiClient.java: commandAction() exit");
          destroyApp(true);
          notifyDestroyed();
    }
    else if (c == start_command )
    {
        show_form(PORT_SCREEN);
    }
    else if (c == back_command)
    {
        show_form(START_SCREEN);
    }
    else if (c == connect_command )
    {
        port = Integer.parseInt(tf.getString());
        proxy = new Proxy(this);
        proxy.start();
    }
  }

  public void show_form(int type)
  {
        form = new Form("form");
        if (type == START_SCREEN)
        {
            form.setTitle("Client Proxy");
            form.addCommand(exit_command);
            form.addCommand(start_command);
            System.out.println("guiClient.java: show_form(START_SCREEN)
called.");
        }
        else if (type == PORT_SCREEN)
        {
            form.setTitle("Set Connection");
            tf = new TextField("Port number:", String.valueOf(DEFAULT_PORT),
                            6, TextField.NUMERIC);
            form.append(tf);
            form.addCommand(back_command);
            form.addCommand(connect_command);
            System.out.println("guiClient.java: show_form(PORT_SCREEN)
called.");
        }
        form.setCommandListener(this);
        display.setCurrent(form);
  }

  public int get_port()
  {
```

```
            return port;
    }
}
/*
 * The purpose of this file is to define a connection class which will
 * make the main user interface look neat.
 *
 * This class sets up the socket connection through which the client
 * device could interact with the server application.
 */

import java.io.*;
import java.io.InputStream;
import java.io.OutputStream;
import javax.microedition.io.*;

public class Connection
{
        public SocketConnection sc;
        public InputStream is;
        public OutputStream os;

        public Connection(int PORT)
        {
                try
                {
                        sc = (SocketConnection)
Connector.open("socket://localhost:"+PORT);
                        if (sc != null)
                        {
                                os = sc.openOutputStream();
                                is = sc.openInputStream();
                        }
                }
                catch (IOException e)
                {
                        System.out.println(e.getMessage());
                }
        }

        public void close()
        {
                try
                {
                        if (is != null)
```

```
                                        is.close();
                              if (os != null)
                                        os.close();
                              if (sc != null)
                                        sc.close();
                    }
                    catch (IOException ioe)
                    {
                    }

          }
}
/*
 * This program is the main thread that handles the search function
 * on the mobile device.
 * It takes the user input as search phrase;
 *    sends the user query to the server side;
 *    lists search document titles from the serve;
 *    displays document description to the mobile user.
 */

import java.io.*;
import java.util.Vector;
import javax.microedition.io.*;
import javax.microedition.lcdui.*;

//import javax.microedition.midlet.*;

// To review the J2ME documents that come with the compiler
// Start -> All Programs -> Sun Java(TM) Wireless Toolkit 2.5.2 for CLDC ->
Documentation.

//Create a J2ME project under Eclipse SDK 3.3.2
//File -> New -> Project... -> J2ME -> J2ME Midlet Suite
//Window "New J2ME Project" -> Project name: -> Next
//Under "Device", "Group:" select "Sun Java(TM) Wireless Toolkit 2.5.2 for CLDC
//            "Device:" select "MediaControlSkin" or other skins provided by the toolkit.
//Click "Next" if I want to import additional libraries such as kxml2-2.3.0.jar

// Note: the request sent to server has to be in the form
// <request>aaa</request><value>bbb</value>
public class Proxy implements Runnable, CommandListener
{
          //The display for this MIDlet
          private Display display;
```

```
        private guiClient parent;
        private Sender sender;
        private Receiver receiver;

        // message_queue: defined to control the communication between threads.
        private Vector message_queue, search_document_id_queue,
search_document_title_queue;

        private SocketConnection sc;
        private InputStream is;
        private OutputStream os;

        private List search_result_List;
        private StringItem error_string;
        private TextField tf;
        private Form form;
        private final int ERROR_SCREEN=0, SEARCH_SCREEN=1,
SEARCH_RESULT_SCREEN=2, SHOW_DOCUMENT_SCREEN=3,
                                        SHOW_DOCUMENT_SUMMARY=4;
        private int port = 0;
        private static boolean connected = false;

        // format: new Command("Title", Type, Priority);
        // "Title": string description
        // Type: Command.ITEM, Command.EXIT, Command.BACK, Command.OK.
        // Priority: The application uses the priority value to describe the importance of
        //          this command relative to other commands on the same screen.
        private Command send_command = new Command("Send", Command.ITEM, 1);
        private Command proxy_back_command = new Command("Back",
Command.BACK, 0);
        private Command back_command = new Command("Back", Command.BACK,
0);
        private Command summary_back_command = new Command("Back",
Command.BACK, 0);
        private Command ok_command = new Command("Ok", Command.OK, 2);
        private Command view_document_command = new Command("View",
Command.ITEM, 0);


        // global variables for document results.
        private String[] document_array;
        private Image[] image_array = null;
        private Command search_back_command = new Command("Back",
Command.BACK, 0);
        //private StringItem document_title, document_summary;
```

```
	private String title, summary;

	/*
	 * Construction
	 */
	public Proxy(guiClient Parent)
	{
		parent = Parent;
		// must get port number.
		port = Parent.get_port();
		display = Display.getDisplay(Parent);
		System.out.println("Proxy->Proxy() called. port: "+port);
	}

	public void show_form(int type)
	{
		form = new Form("form");
		if (type == ERROR_SCREEN)
		{
			form.setTitle("Alert");
			error_string = new StringItem("Error:", " ");
			form.append(error_string);
			form.addCommand(back_command);
			form.addCommand(ok_command);
			System.out.println("Proxy->show_form(ERROR_SCREEN)
called.");
			form.setCommandListener(this);
			display.setCurrent(form);
		}
		else if (type == SEARCH_SCREEN)
		{
			form.setTitle("Proxy");
			tf = new TextField("Search Phrase:", "", 20, 1);
			form.append(tf);
			form.addCommand(proxy_back_command);
			form.addCommand(send_command);
			System.out.println("Proxy->show_form(SEARCH_SCREEN)
called.");
			form.setCommandListener(this);
			display.setCurrent(form);
		}
		else if (type == SEARCH_RESULT_SCREEN)
		{
		search_result_List = new List("Exclusive", Choice.IMPLICIT,
document_array,image_array);
```

```
                search_result_List.setCommandListener(this);
                search_result_List.addCommand(search_back_command);
                search_result_List.addCommand(view_document_command);
                display.setCurrent(search_result_List);
                }
                else if (type == SHOW_DOCUMENT_SUMMARY)
                {
                        //form.deleteAll();
                        //form.setTitle("Document");
                        //form.removeCommand(proxy_back_command);
                        //form.removeCommand(send_command);

                        StringItem title = new StringItem(this.title, " ");
                        form.append(title);
                        form.append(this.summary);
                        form.addCommand(summary_back_command);
                        form.setCommandListener(this);
                        display.setCurrent(form);
                }
        }


        public void start()
        {
        Thread t = new Thread(this);
        t.start();
   }

        // function must be synchronized for "wait()";
        public void run()
        {
                show_form(SEARCH_SCREEN);

                String message = null;

                message_queue = new Vector();

                set_connection();

                // kick off the receiver thread.
                // message_queue is shared between this thread and the receiver thread.
                // message_queue is synchronized to make sure the communication
between
                // two threads, because this thread and receiver thread are trying to
                // access message_queue. Synchronization makes sure each thread will
                // wait until the other locking thread to release the lock.
                // Once the thread has the access to the message_queue, it will lock it
                // and thus block the other thread to access to it.
                receiver = new Receiver(is, message_queue);

                while (connected)
                {
                        message = "";
                        synchronized(message_queue)
                        {
                                if (message_queue.size()>0)
                                {
                                        message = (String) message_queue.elementAt(0);
                                        message_queue.removeElementAt(0);
                                        //System.out.println("Proxy->run():
message_queue: "+message_queue.size());
                                        if (message != "")
                                        {
                                                System.out.println("Proxy->run():
"+message);
                                                if (message.indexOf("documents:")>=0)
                                                {
                                                        showSearchResults(message);
                                                }
                                                else if (message.indexOf("Summary:")>=0)
                                                {
                                                        showDocumentSummary(message);
                                                }
                                        }
                                }
                                else
                                {
                                        try
                                        {
                                                // put the thread in hold mode.
                                                message_queue.wait();
                                        }
                                        catch (InterruptedException e)
                                        {
                                                System.out.println("run():
"+e.getMessage());
                                        }
                                }
                        }
                }
```

```
	}

	public void commandAction(Command c, Displayable d)
	{
		String phrase;
		if (c == back_command)
	  {
			// back to parent screen
			System.out.println("Proxy->commandAction() back to client.");
			parent.show_client();
	  }
		else if (c == send_command)
		{
			phrase = tf.getString();
			if (phrase != null)
			{
				System.out.println("Proxy->commandAction() send
\""+phrase+"\"");

	sender.sendMessage("<request>search</request><value>"+phrase+"</value>");
			}
		}
		else if (c == proxy_back_command)
		{
			stop();
			parent.show_client();
		}
		else if (c == search_back_command)
		{
			show_form(SEARCH_SCREEN);
		}
		else if (c == summary_back_command)
		{
			buildDocumentScreen();
		}
		else if (c == List.SELECT_COMMAND || c ==
view_document_command)
		{
			if (d.equals(search_result_List))
			{
				requestDocumentSummary();
			}
		}
	}
	/*
	 * This function sends the user selected document id to the server.
	 * The source user interface is the SEARCH_RESULT_SCREEN which shows
	 * user the returned documents for the submitted query.
	 * When server returns the selected document description, the while
	 * loop in the run() will catch such action and switch to
showDocumentDescription().
	 */
	private void requestDocumentSummary()
	{
		int document_index = search_result_List.getSelectedIndex();
		String document_id =
(String)search_document_id_queue.elementAt(document_index);
		System.out.println("Proxy->requestDocumentSummary() document id:
"+document_id);

	sender.sendMessage("<request>document</request><value>"+document_id+"</v
alue>");
	}

	private void set_connection()
	{
		try
		{
			sc = (SocketConnection)
Connector.open("socket://localhost:"+port);
			if (sc != null)
			{
				os = sc.openOutputStream();
				sender = new Sender(os);
				is = sc.openInputStream();
				connected = true;
			}
		}
		catch (IOException e)
		{
			System.out.println(e.getMessage());
		}
	}

   public void stop()
   {
	try
	{
```

```
                    synchronized (message_queue)
                    {
                              if (sender != null)
                                        sender.stop();

                              if (is != null)
                                        is.close();

                              if (os != null)
                                        os.close();

                              if (sc != null)
                                        sc.close();

                              connected = false;
                    }
          }
          catch (IOException e)
          {
          }
   }

   // At the end, the user interface will be changed via call
show_form(SEARCH_RESULT_SCREEN).
   private void buildDocumentScreen()
   {
          int d;
          document_array = new String[search_document_id_queue.size()];
          image_array = null;

          for (d=0; d<search_document_id_queue.size(); d++)
          {
                    document_array[d] = (String) search_document_title_queue.elementAt(d);
                    System.out.println("Proxy-
>buildDocumentScreen():"+search_document_title_queue.elementAt(d));
          }
          show_form(SEARCH_RESULT_SCREEN);
   }


   // argument is in the format
   // Delimiter = |||
   // documents:document_id, document_title|||document_id, document_title|||...
document_id, document_title
```

```
    private void showSearchResults(String message)
    {
            search_document_id_queue = new Vector();
            search_document_title_queue = new Vector();

            String temp, document="", Delimiter = "|||";
            // remove "documents:"
            temp = message.substring("documents:".length());

            String document_id, document_title;
            //////// My Version's of String.split ///////
        while(temp.indexOf(Delimiter)>=0)
        {
            document = temp.substring(0, temp.indexOf(Delimiter));
            temp = temp.substring(temp.indexOf(Delimiter)+Delimiter.length(),
temp.length());

            document_id = document.substring(0, document.indexOf(","));
            document_title = document.substring(document.indexOf(",")+1,
document.length());
            search_document_id_queue.addElement(document_id);
            search_document_title_queue.addElement(document_title);
            //System.out.println("Proxy->showSearchResults():id: "+document_id+", title:
"+document_title);
        }
        // catch the last one
        if (temp.length()>0)
        {
            document_id = temp.substring(0, temp.indexOf(","));
            document_title = temp.substring(temp.indexOf(",")+1, temp.length());
            search_document_id_queue.addElement(document_id);
            search_document_title_queue.addElement(document_title);
            //System.out.println("Proxy->showSearchResults():id: "+document_id+", title:
"+document_title);
        }
        ////////My Version's of String.split ///////

        buildDocumentScreen();
    }


    private void showDocumentSummary(String message)
    {
                int document_index = search_result_List.getSelectedIndex();
```

```
                    String document_id =
(String)search_document_id_queue.elementAt(document_index);

                    // get document title
                    title = (String)search_document_title_queue.elementAt(document_index);

                    // remove "description:"
                    summary = message.substring("Summary:".length());

                    System.out.println("Proxy->showDocumentSummary() document:
"+summary);
                    show_form(SHOW_DOCUMENT_SUMMARY);
    }
}
/*
 * This class provides XML extraction for the client device.
 * When XML data is received from the server, receiver parses
 * the XML data and pass to the client application.
 * All messages from server are in XML format.
 */

import java.io.*;
import java.util.Vector;

/*
import javax.xml.parsers.DocumentBuilderFactory;
import javax.xml.xpath.*;
import org.w3c.dom.*;
*/
//import javax.microedition.midlet.*;

//To review the J2ME documents that come with the compiler
//Start -> All Programs -> Sun Java(TM) Wireless Toolkit 2.5.2 for CLDC ->
Documentation.

// Create a J2ME project under Eclipse SDK 3.3.2
// File -> New -> Project... -> J2ME -> J2ME Midlet Suite
// Window "New J2ME Project" -> Project name: -> Next
// Under "Device", "Group:" select "Sun Java(TM) Wireless Toolkit 2.5.2 for CLDC
//                "Device:" select "MediaControlSkin" or other skins provided by the toolkit.
// Click "Next" if I want to import additional libraries such as kxml2-2.3.0.jar

// Approach one (I take this approach)
// Install kxml parser with Eclipse SDK 3.3.2
// 1) Download archive file kxml2-2.3.0.jar from Internet.
// 2) Right click on project "guiClient", and then click on "Properties".
// 3) Select "Java Build Path" in the left pan, and click on "Libraries" tag.
// 4) Click on "Add External JAR..."
// 5) Select kxml2-2.3.0.jar from the source folder.

// Approach two
//To compile this project with Sun Java Wireless Toolkit 2.5.2
//1) Creating project "guiClient"
//2) Place Java source files in "C:\Documents and
Settings\feng\j2mewtk\2.5.2\apps\guiClient\src"
//3) Place application resource files in "C:\Documents and
Settings\feng\j2mewtk\2.5.2\apps\guiClient\res"
//4) Place application library files (kxml2-2.3.0.jar) in "C:\Documents and
Settings\feng\j2mewtk\2.5.2\apps\guiClient\lib"
//5) Settings updated
//6) Project settings saved

import org.kxml2.io.*;
import org.xmlpull.v1.*;

public class Receiver extends Thread
{
  private StringBuffer sb;
  private Vector message_queue;
  private String message;
        private InputStream is;
        private KXmlParser parser;
        private Vector frame_list;

        /*
         * Constructor starts thread automatically.
         */
  public Receiver(InputStream is, Vector message_queue)
  {
        this.is = is;
        sb = new StringBuffer();
        this.message_queue = message_queue;
        frame_list = new Vector();
        start();
  }


  public void run()
  {
        int c;
```

```
          while(true)
          {
                    message = "";

                    try
                    {
                              // Remember string is appended with '\n' to exit the while loop.
                              while (((c = is.read()) != '\n') && (c != -1))
                              {
                                        sb.append((char) c);
                              }
                    }
                    catch (IOException e)
                    {
                    }
                    message = sb.toString();


                    synchronized(message_queue)
                    {
                              System.out.println("Receiver->run() received
message:"+message);
                              if (message != "")
                              {
                                                  message = extract(message);
                                        //message = extractXML(message);
                                                  if (message != "")
                                                  {
                                                            System.out.println("Receiver->run()
extracted message:"+message+", frame_list size:"+frame_list.size());
                                                            message_queue.addElement(message);
                                                            message_queue.notifyAll();
                                                  }
                              }
                    }

          }
   }

   /*
    *  This function utilizes KXmlParser to extract new message from the repeated
message.
    *  The reason why I receive repeated message is not known.
    *  The trick is to call parser.nextTag() and parser.nextText() hand in hand.
    *  parser.nextTag() moves to the next xml tag, while parser.nextText() extract the
message
    *  between the xml tags.
    */
   public String extract(String xml_data)
   {
         String name, value="";

         // The following three lines are necessary to create xml parser.
         byte[] xml_byte_array = xml_data.getBytes();
         ByteArrayInputStream xml_stream = new
ByteArrayInputStream(xml_byte_array);
         InputStreamReader xml_reader = new InputStreamReader(xml_stream);

         boolean extracted = false;

         try
         {
                parser = new KXmlParser();
                parser.setInput(xml_reader);

                while (parser.nextTag()!= XmlPullParser.END_TAG)
                {
                       parser.require(XmlPullParser.START_TAG, null, null);
                       name = parser.getName();
                       value = parser.nextText();

                       if (name.equals("time"))
                       {
                       if (frame_list.isEmpty() || frame_list.indexOf(value)< 0)
                       {
                              //System.out.println("checkDuplication(): frame_list
size:"+frame_list.size());
                              frame_list.addElement(value);
                              extracted = true;
                       }
                       }
                       else if (name.equals("load"))
                       {
                              //Note: tag <load> comes after <time>
                              if (extracted)
                                     break;
                       }

                parser.require(XmlPullParser.END_TAG, null, name);
                }
                parser.require(XmlPullParser.END_DOCUMENT, null, null);
        }
        catch (Exception e)
        {
                System.out.println("Receiver.java: extract() "+e.getMessage());
        }
        return value;
   }

/*
   public String extractXML(String xml_data)
   {
        String value = "";
        String time, load;
        try
        {
                InputStream is = new ByteArrayInputStream(xml_data.getBytes("UTF-
8"));
                Document response =
DocumentBuilderFactory.newInstance().newDocumentBuilder().parse(is);

                XPathFactory factory = XPathFactory.newInstance();
        XPath xPath=factory.newXPath();

        NodeList nodes = (NodeList)xPath.evaluate("/message", response,
XPathConstants.NODESET);
        int nodeCount = nodes.getLength();

        //iterate over search Result nodes
        for (int i = 0; i < nodeCount; i++)
        {
           //Get each xpath expression as a string
           time = (String)xPath.evaluate("time", nodes.item(i), XPathConstants.STRING);
           load = (String)xPath.evaluate("load", nodes.item(i), XPathConstants.STRING);

           //print out the Title, Summary, and URL for each search result
           System.out.println("time: " + time);
           System.out.println("load: " + load);

           value += load;
        }

        }
        catch (IOException e)
```

```
          {

          }
          catch (Exception e)
          {

          }
          return value;
    }
*/
}
/*
 * This class sends message to the server side.
 * Messages are in the queue.
 */

import java.io.*;

/*
 * Thread: The other way to create a thread is to declare a class that implements
 *         the Runnable interface. That class then implements the run method.
 * CommandListener: Make sure object of this class could listen to "Command".
 *                 Must implement
 */
public class Sender extends Thread
{
   private String message;
          private OutputStream os;

          /*
           * Constructor starts thread automatically.
           */
   public Sender(OutputStream os)
   {
          this.os = os;
          start();
   }

   /*
    * stop() forces the thread to finish, no more wait.
    */
   public synchronized void stop()
   {
      message = null;
      notify();
```

```
  }

  public synchronized void sendMessage(String msg)
  {
    message = msg;
    // wake up the thread by itself.
    System.out.println("Sender.java: "+msg);
    notify();
  }

  public synchronized void run()
  {
        while(true)
        {
                // Constructor calls run() and message is null, wait for the message.
        if (message == null)
        {
          try
          {
                wait();
          }
          catch (InterruptedException e)
          {
                System.out.println("Sender.java: run() "+e.getMessage());
          }
        }

        // if this thread is reactivated [by notify()], then make sure message is not null.
        // Or stop() must be called to gracefully finish the thread.
        if (message == null)
        {
                System.out.println("Sender.java: run() thread is notified with message set
to null");
                break;
        }

        try
        {
                os.write(message.getBytes());
          os.write("\r\n".getBytes());
        }
        catch (IOException ioe)
        {
          ioe.printStackTrace();
        }
```

```
          // After the message is sent, clear message and go back to the top of the while
loop
          // which means the thread is back to waiting mode.
          message = null;
       }
    }

}
```

Appendix C. Server Side Implementation Code

```
/*
*
* author: Feng Gui
*
* Description:
*
* This is the main program at the server.
* It listens to the IP port. If it detects the mobile user request,
* it will generate a thread to deal with this particular mobile user.
* This program is to simulate the server side reaction.
*/

import java.net.*;
import java.io.*;
import java.awt.*;
import javax.swing.*;
import javax.swing.border.*;

public class Manager
{
  static final int FRAME_WIDTH=700, FRAME_HEIGHT=300,
BORDER_OFFSET=50;

  public static void main(String[] args) throws IOException
  {
    // Set up the JFrame.  The JFrame dimensions are the same as the ones of the canvas.
    JFrame.setDefaultLookAndFeelDecorated(true);
    JFrame jframe;
    jframe = new JFrame("Intelligent Manager");
    jframe.setSize(FRAME_WIDTH, FRAME_HEIGHT);
    jframe.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    jframe.getContentPane().setLayout(new BorderLayout());

    //jframe.setIconImage(new
ImageIcon(jframe.getClass().getResource("resources/icons/RSS_004.png")).getImage());
    Image icon = Toolkit.getDefaultToolkit().getImage("icons/RSS_001.png");
    jframe.setIconImage(icon);
    //jframe.setIconImage(new
ImageIcon(jframe.getClass().getResource("icons/RSS_004.png")).getImage());
    centerFrame(jframe);

    // user interface components.
    JEditorPane jep;
    JScrollPane jsp;

    // Create the text area.
    jep = new JEditorPane();
    jep.setEditable(false);
    jsp = new JScrollPane(jep);
    jsp.setPreferredSize(new Dimension(jframe.WIDTH-BORDER_OFFSET,
jframe.HEIGHT-BORDER_OFFSET));
    jframe.getContentPane().add(jsp, BorderLayout.CENTER);


    // Create button panel at bottom.
    JPanel bottomPanel = new JPanel();
    Dimension buttonSize = new Dimension(80,20);
    JButton closeButton = new JButton("Close");
    closeButton.setPreferredSize(buttonSize);
    closeButton.addMouseListener(new java.awt.event.MouseAdapter()
                    {
                      public void mouseClicked(java.awt.event.MouseEvent evt)
                      {
                        System.exit(1);
                      }
                    });
    bottomPanel.add(closeButton);
    bottomPanel.setBorder(new
CompoundBorder(BorderFactory.createLineBorder(Color.LIGHT_GRAY, 2),
BorderFactory.createBevelBorder(BevelBorder.LOWERED)));
    jframe.getContentPane().add(bottomPanel, BorderLayout.SOUTH);

    jframe.setVisible(true);
    ServerSocket serverSocket = null;
    boolean listening = true;

    int count=1, port=8888;

    try
    {
      serverSocket = new ServerSocket(port);
      jep.setText("MyServer is up and listening.\n");
        System.out.println("\nMyServer is up and listening.\n");
    }
    catch (IOException e)
    {
      System.err.println("Could not listen on port: "+port+".\n");
      System.exit(-1);
      }

      while (listening)
      {
        //new guiThread(serverSocket.accept()).start();
        jep.setText(jep.getText()+"\nThread "+count+" is created.\n");
        System.out.println("Thread "+count+" is created.");
        new Thread(new MyThread(serverSocket.accept(), count)).start();
        count++;
      }

      serverSocket.close();
    }

    private static void centerFrame(JFrame jframe)
    {
      // Center the main frame only if the frame size is smaller than the window screen.
      Toolkit windowScreen = jframe.getToolkit();
      Dimension windowSize = windowScreen.getScreenSize();
      if (FRAME_WIDTH < windowSize.width && FRAME_HEIGHT <
windowSize.height)
        jframe.setBounds((windowSize.width-FRAME_WIDTH)/2, (windowSize.height-
FRAME_HEIGHT)/2, FRAME_WIDTH, FRAME_HEIGHT);
      else
        jframe.setBounds(0,0,windowSize.width,windowSize.height);
    }
}
/*
 * MySQL.java
 *
 * Created on January 13, 2007, 1:12 PM
 *
 * This is the foundation class that provides the database connection to
 * MySQL server. It provides connection and statement objects to the
 * other functions.
 */

import java.sql.*;

/*
 * Copy mysql-connector-java-5.1.6-bin.jar to C:\Program
Files\Java\jdk1.6.0_10\jre\lib\ext
 */
public class MySQL
{
    public Statement getConnectionStatement() throws SQLException,
ClassNotFoundException
    {
      Class.forName(mysqlDriver);
      mysqlConnection = DriverManager.getConnection(mysqlConnectionString,
mysqlUser, mysqlPassword);
      return mysqlConnection.createStatement();
    }

    public Connection getConnection() throws SQLException, ClassNotFoundException
    {
      Class.forName(mysqlDriver);
      mysqlConnection = DriverManager.getConnection(mysqlConnectionString,
mysqlUser, mysqlPassword);
      return mysqlConnection;
    }

    /** Creates a new instance of MySQL */
    public MySQL()
    {
      mysqlDriver = "com.mysql.jdbc.Driver";
      mysqlConnectionString = "jdbc:mysql://localhost:3306/vendor";
      mysqlUser = "feng";
      mysqlPassword = "fenggui";
    }

    // Mysql Login Information.
    private String mysqlDriver, mysqlConnectionString, mysqlUser, mysqlPassword;
    private Connection mysqlConnection;
}
/*
 * This is the thread that instantiated by the listening server program upon request from
 * mobile user.
 *
 * This thread interacts with mobile device to provide services requested.
 * The user interface is a frame that monitors the activities between mobile user and
server.
 * User requests are structured in XML format.
 * User requests are extracted from XML data and search query are further processed by
 * search manager. Search results are sent back to the mobile user in XML format.
 * The core functions are direct search with the Yahoo search engine.
 * The other one is the context-aware search.
 */

import java.net.*;
import java.io.*;
import java.sql.*;
import java.awt.*;
import javax.swing.*;
import javax.swing.border.*;
import java.util.*;

// import Kxml parser
// 1) set up a Java Application project in NetBeans IDE 6.1
// 2) Right click on "Libraries" and select "Add JAR/Folder..."
// 3) When "Add JAR/Folder" window pops up, find in the right folder/path and
//    select the right file "kxml2-2.3.0.jar".
import org.kxml2.io.*;
import org.xmlpull.v1.*;

public class MyThread implements Runnable
{
   // flag for debug purpose. Set to true, print_debug() will output lines for debug.
   private boolean debug = true;

   // variables for communication with client.
   private Socket socket = null;
   private PrintWriter to_client;
   private BufferedReader from_client;
   private String client_message, server_message;
   private int thread_order;
   private int Message_Count = 0, SEARCH_TYPE, DIRECT_SEARCH=1,
CONTEXT_SEARCH=2;

   // Third-party XML parser.
   private KXmlParser parser;

   // variable defined to carry the extracted XML message sent from client.
   // xml format: <request>xxx</request><value>yyy</vaule>.
   private Request request;

   // user interface components.
   private JFrame jframe;
   JEditorPane jep;
   JScrollPane jsp;
   static final int FRAME_WIDTH=700, FRAME_HEIGHT=300,
BORDER_OFFSET=50;

   // SearchManager interacts with Yahoo search API to get query results for
   // 1) direct search (no user reference frame merged with user query)
```

```
  // 2) context search (user query is merged with user reference frame)
  private SearchManager search_manager;

  public MyThread(Socket socket, int ThreadCount)
  {
    // socket is passed from server.
    this.socket = socket;
    thread_order = ThreadCount;

    search_manager = new SearchManager();

    try
    {
      to_client = new PrintWriter(socket.getOutputStream(), true);
        from_client = new BufferedReader(new
InputStreamReader(socket.getInputStream()));
    }
    // Catch socket error.
    catch (IOException e)
    {
      e.printStackTrace();
    }
    print_debug("MyThread: MyThread() is construced.");
  }

  public void run()
  {
    launchFrame();

    boolean connected = login();
    boolean sent = false;
    print_debug("MyThread.java->run(): connected = "+connected);


    while (connected)
    {
      client_message = receive();

      if (client_message != null)
      {
        request = extractRequest(client_message);
        print_debug("MyThread.java->run(): detected
request:"+request.getTagName()+", value:"+request.getTagValue());
      }
```

```
        if (request.getTagName().equals("search"))
        {
           // approach 1) direct search
           //search_manager.getSearchResults(request.getTagValue());
           //SEARCH_TYPE = DIRECT_SEARCH;
           //send(search_manager.getSearchResultString());

           // approach 2) context search
           search_manager.getContextSearchResults(request.getTagValue());
           SEARCH_TYPE = CONTEXT_SEARCH;
           // sort returned search documents based on context profile.
           search_manager.orderSearchResult();
           send(search_manager.getSortedSearchResultString());
        }

        if (request.getTagName().equals("document"))
        {
           sendRequestedDocumentSummary(Integer.parseInt(request.getTagValue()));
        }

      }

     print_debug("MyThread.java: run() thread finished.");

    }


    private void sendRequestedDocumentSummary(int document_id)
    {
       String summary = "";

       if (SEARCH_TYPE == DIRECT_SEARCH)
          summary = search_manager.getSearchDocumentSummary(document_id,
DIRECT_SEARCH);
       else if (SEARCH_TYPE == CONTEXT_SEARCH)
          summary = search_manager.getSearchDocumentSummary(document_id,
CONTEXT_SEARCH);

       System.out.println("MyThread.java->sendRequestedDocument() "+summary+",
document_id: "+document_id);

       send(summary);
    }
```

```
  private boolean login()
  {
    boolean done = false;

    print_debug("login(): A new thread is running.");

    // Receive client login information
    //clientMessage = receive();
    //print_debug("MyThread.java(): login(): "+clientMessage);

    send("welcome to context server");
    done = true;
    return done;
  }


  private void launchFrame()
  {
    // Set up the JFrame. The JFrame dimensions are the same as the ones of the canvas.
    JFrame.setDefaultLookAndFeelDecorated(true);
    jframe = new JFrame("Thread "+thread_order);
    jframe.setSize(FRAME_WIDTH, FRAME_HEIGHT);
    jframe.getContentPane().setLayout(new BorderLayout());
    //jframe.setIconImage(new
javax.swing.ImageIcon(jframe.getClass().getResource("RSS_002.png")).getImage());

    // Create the text area.
    jep = new JEditorPane();
    jep.setEditable(false);
    jsp = new JScrollPane(jep);
    jsp.setPreferredSize(new Dimension(jframe.WIDTH-BORDER_OFFSET,
jframe.HEIGHT-BORDER_OFFSET));
    jframe.getContentPane().add(jsp, BorderLayout.CENTER);

    // Create button panel at bottom. The button panel contains only one button.
    JPanel bottomPanel = new JPanel();
    Dimension buttonSize = new Dimension(80,20);
    JButton closeButton = new JButton("Close");
    closeButton.setPreferredSize(buttonSize);
    closeButton.addMouseListener(new java.awt.event.MouseAdapter()
                        {
                          public void mouseClicked(java.awt.event.MouseEvent evt)
                          {
                            System.exit(1);
                          }
```

```
                    });
    bottomPanel.add(closeButton);

    bottomPanel.setBorder(new
CompoundBorder(BorderFactory.createLineBorder(Color.LIGHT_GRAY, 2),
BorderFactory.createBevelBorder(BevelBorder.LOWERED)));
    jframe.getContentPane().add(bottomPanel, BorderLayout.SOUTH);

    Image icon = Toolkit.getDefaultToolkit().getImage("icons/RSS_002.png");
    jframe.setIconImage(icon);
    centerFrame(jframe);
    jframe.setVisible(true);
  }


  private static void centerFrame(JFrame jframe)
  {
     // Center the main frame only if the frame size is smaller than the window screen.
     Toolkit windowScreen = jframe.getToolkit();
     Dimension windowSize = windowScreen.getScreenSize();
     if (FRAME_WIDTH < windowSize.width && FRAME_HEIGHT <
windowSize.height)
        jframe.setBounds((windowSize.width-FRAME_WIDTH)/2, (windowSize.height-
FRAME_HEIGHT)/2, FRAME_WIDTH, FRAME_HEIGHT);
     else
        jframe.setBounds(0,0,windowSize.width,windowSize.height);
  }


  // Request is defined class which specify the request type/value pair from
  // client.
  private Request extractRequest(String message)
  {
    // variables defined for processing xml.
    parser = new KXmlParser();
    byte[] xml_byte_array = message.getBytes();
    ByteArrayInputStream xml_stream = new ByteArrayInputStream(xml_byte_array);
    InputStreamReader xml_reader = new InputStreamReader(xml_stream);

    String xml_tag="", value="";
    Request request = new Request();

    try
    {
     parser.setInput(xml_reader);
```

```
      while (parser.nextTag()!= XmlPullParser.END_TAG)
      {
      parser.require(XmlPullParser.START_TAG, null, null);
      xml_tag = parser.getName();
      value = parser.nextText();

      if (xml_tag.equals("request"))
      {
      request.setTagName(value);
      }
      else if (xml_tag.equals("value"))
      {
         request.setTagValue(value);
      }
      parser.require(XmlPullParser.END_TAG, null, xml_tag);
      }
   parser.require(XmlPullParser.END_DOCUMENT, null, null);
  }
  catch (XmlPullParserException e)
  {
  }
  catch (IOException e)
  {
  }

  return request;
}


private void send(String line)
{
  try
  {
   to_client = new PrintWriter(socket.getOutputStream(), true);
  }
  // Catch socket error.
  catch (IOException e)
  {
    e.printStackTrace();
  }
  if (line.length()>0)
  {
    to_client.println("<time>"+Message_Count+"</time><load>"+line+"</load>\r\n");
    print_to_frame("\nSent: "+line+"\n");
```

```
      Message_Count++;
    }
  }

  private String receive()
  {
    String message = new String("");
    try
    {
      if ( (message = from_client.readLine()) != null )
         message = message.trim();
    }
    catch (IOException e)
    {
      e.printStackTrace();
      System.exit(0);
    }

    if (message.length()>0)
    {
       print_to_frame("\nReceived: "+message+"\n");
       print_debug("MyThread->receive():"+message);
    }
    return message;
  }

  private void print_to_frame(String line)
  {
    if (line.length()>0)
    {
       jep.setText(jep.getText()+line);
    }
  }

  private void print_debug(String line)
  {
    if (debug)
      System.out.println(line);
  }
}

/*
 * Query manager handles user search query with user context.
 * User reference frame is processed and user context is studied.
 * User context is merged with user query.
 */

import java.sql.*;
import javax.swing.*;
import java.util.*;

/**
 *  QueryManager deals with query augmentation.
 *  User query is merged with user reference frame.
 **/
public class QueryManager
{
  // MySQL database class.
  private MySQL mysql;
  private Statement statement;
  //private String augmented_user_query;

  // This vector has all the user queries
  private Vector<Data> user_query;

  // all reference frames sent from client.
  private ReferenceFrame reference_frames;

  // This variable stores the augmented profile which is set in function
setAugmentedProfile()
  // If the user query exists (sent from client), then the user query is merged into it based
  // on the tf-idf score in descending order.
  // If the user query is missing, then the last reference frame is used.
  private ReferenceFrame augmented_reference_frame;

  public QueryManager()
  {
     mysql = new MySQL();

     // Each user query word is stored as a unit in the vector userQuery.
     user_query = new Vector<Data>();

     // QueryManager constructor loads user reference frames from database.
     reference_frames = new ReferenceFrame();
     loadReferenceFrames();
  }

  // This function reads user query from the argument.
  public void setUserQuery(String line)
  {
    if (line == null || line.length() == 0)
      return;

    int q, iteration=1;
    // User input
    int SOURCE = 2;
    String[] queryArray = line.trim().split(" ");
    for (q=0; q<queryArray.length; q++)
      user_query.add(new Data(queryArray[q], SOURCE, iteration));
  }


  /**
   * One important thing I need to remember is that the context weights in the
   * reference frame are sorted in the query that retrieves the reference frames.
   * Context weights refer to value in the sql command.
   * This function loads all the reference frames from the database.
   **/
  private void loadReferenceFrames()
  {
    try
    {
      statement = mysql.getConnectionStatement();
      String sql = "select * from reference order by iteration asc, value desc";
      ResultSet rs = statement.executeQuery(sql);
      while (rs.next())
      {
        reference_frames.addReferenceData(rs.getString("type"),
Double.valueOf(rs.getString("value")), rs.getInt("source"), rs.getInt("frequency"),
rs.getInt("iteration"), rs.getInt("index"));
      }
      statement.close();
    }
    catch(ClassNotFoundException e)
    {
      JOptionPane.showMessageDialog(null, "Mysql driver class not found.", "Alert",
JOptionPane.ERROR_MESSAGE);
    }
    catch(SQLException e)
    {
      JOptionPane.showMessageDialog(null, e.getMessage(), "Alert",
JOptionPane.ERROR_MESSAGE);
      //e.printStackTrace();
    }
  }
  public String getLocation()
  {
    PreparedStatement ps;
    Connection connection;

    String location = "";

    try
    {
      connection = mysql.getConnection();
      ps = connection.prepareStatement("select * from location where iteration = (select
max(iteration) from location)");
      ResultSet rs = ps.executeQuery();

      if (rs.next())
        location = rs.getString("location");

      connection.close();
    }
    catch(ClassNotFoundException ex)
    {
      JOptionPane.showMessageDialog(null, "Mysql driver class not found.", "Alert",
JOptionPane.ERROR_MESSAGE);
    }
    catch(SQLException ex)
    {
      JOptionPane.showMessageDialog(null, ex.getMessage(), "Alert",
JOptionPane.ERROR_MESSAGE);
      //e.printStackTrace();
    }

    return location;
  }

  /**
   * This function merges reference frame and user query into augmented context profile
   * The resulted aReferenceFrame should contain merged user query.
   **/
  public void setAugmentedProfile()
  {
    ReferenceData rd;

    int i, lastIteration = reference_frames.getLastIteration();
    // Right now only work on the last iteration of the reference frame.
    if (lastIteration > 0)
      augmented_reference_frame =
reference_frames.getOneReferenceFrame(lastIteration);

    // Merge user query if it is not empty
    if (!user_query.isEmpty())
    {
      // If the reference frame has the user query word, increase weight for that context
entity
      // by max weight in that referece frame.
      double maxWeight =
augmented_reference_frame.getMaxWeightInIteration(lastIteration);

      for (i=0; i<user_query.size(); i++)
      {
        rd = augmented_reference_frame.search(user_query.get(i).getType());
        if (rd != null)
        {
          // Make sure the updated context weight is the max in the iteration.
          augmented_reference_frame.updateReferenceData(rd, maxWeight);
        }
        // The reference frame does not have the user query word, insert user query word
        else
        {
          // Currently, I will make the user query the most important weight by
multiplying
          // a factor of 1.5 to the max weight in that iteration.
          double multiple = 1.5;
          // Initialize frequency = 1, and index = 0. Both of them will be updated in
aReferenceFrame.insertWeightInIteration();
          rd = new ReferenceData(user_query.get(i).getType(), maxWeight*multiple,
user_query.get(i).getSource(), 1, lastIteration, 0);
          augmented_reference_frame.insertDataInIteration(lastIteration, rd);
        }
      }
    }

  }

  public String getContextQuery(boolean feed_yahoo_server)
  {
    String query = "", location = getLocation();
    if (augmented_reference_frame != null)
```

```
        query =
augmented_reference_frame.getReferenceData(augmented_reference_frame.getLastIterat
ion(), location, feed_yahoo_server);
      return query;
   }

   public String printAllRefereceFrames()
   {
      String data = "";
      if (reference_frames != null)
      {
         data = reference_frames.printAllReferenceFrames();
      }
      return data;
   }
}
/*
 * ReferenceData.java
 *
 * Created on January 16, 2007, 10:00 PM
 *
 * This is the base class that will be used for reference frame.
 */


/**
 *
 * @author Feng
 */
import java.util.*;

public class ReferenceData extends ClientData
{
   private String timestamp;

   /** Creates a new instance of ReferenceData */
   public ReferenceData(String type, double value, int source, int frequency, int iteration,
int index)
   {
      super(type, value, source, frequency, iteration, index);
      Calendar calendar = Calendar.getInstance();
      timestamp = calendar.get(Calendar.YEAR)+"-
"+(calendar.get(calendar.MONTH)+1)+"-"+calendar.get(calendar.DATE)+"
"+calendar.get(calendar.HOUR)+":"+calendar.get(calendar.MINUTE)+":"+calendar.get(
calendar.SECOND);
```

```
  }

  public ReferenceData(ReferenceData rd)
  {
    super(rd.getType(), rd.getValue(), rd.getSource(), rd.getFrequency(), rd.getIteration(),
rd.getIndex());
    Calendar calendar = Calendar.getInstance();
    timestamp = calendar.get(Calendar.YEAR)+"-
"+(calendar.get(calendar.MONTH)+1)+"-"+calendar.get(calendar.DATE)+"
"+calendar.get(calendar.HOUR)+":"+calendar.get(calendar.MINUTE)+":"+calendar.get(
calendar.SECOND);
  }

  public ReferenceData(ClientData cd)
  {
    super(cd.getType(), cd.getValue(), cd.getSource(), cd.getFrequency(),
cd.getIteration(), cd.getIndex());
    Calendar calendar = Calendar.getInstance();
    timestamp = calendar.get(Calendar.YEAR)+"-
"+(calendar.get(calendar.MONTH)+1)+"-"+calendar.get(calendar.DATE)+"
"+calendar.get(calendar.HOUR)+":"+calendar.get(calendar.MINUTE)+":"+calendar.get(
calendar.SECOND);
  }

  public String getTimestamp()
  {
    return timestamp;
  }
}
/*
 * ReferenceFrame.java
 *
*
 * This is the foundation class that serves the reference frame.
 * Context data is saved into reference frame.
 */
import java.util.*;


/**
 *
 * Class Engine will get all reference frames from database not this class.
 */
public class ReferenceFrame
{
  // According to the algorithm, n refers to the number of effective virtual frames that
  // are used to calcualte the reference frame.
  private int n;
  private Vector<ReferenceData> reference_data;
  private long timestamp;

  /** Creates a new instance of ReferenceFrame */
  public ReferenceFrame()
  {
    n = 1;
    reference_data = new Vector<ReferenceData>();
  }

  /* This is scenario 1 where all the sequential virtual frames have the same
   * context weights.  Hence, n = n(previous) + 1. */
  public void updateN()
  {
    n++;
  }

  public ReferenceData getWeight(int iteration, int index)
  {
    if (!reference_data.isEmpty())
    {
      for (int i=0; i<reference_data.size(); i++)
      {
        if (reference_data.get(i).getIteration() == iteration &&
reference_data.get(i).getIndex() == index)
        {
          return reference_data.get(i);
        }
      }
    }
    return null;
  }

  public double getWeightValue(int iteration, int index)
  {
    if (!reference_data.isEmpty())
    {
      for (int i=0; i<reference_data.size(); i++)
      {
        if (reference_data.get(i).getIteration() == iteration &&
reference_data.get(i).getIndex() == index)
        {
          return reference_data.get(i).getValue();
        }
      }
    }
    return -9999999;
  }

  public String getReferenceFrameWord(int iteration, int index)
  {
    if (!reference_data.isEmpty())
    {
      for (int i=0; i<reference_data.size(); i++)
      {
        if (reference_data.get(i).getIteration() == iteration &&
reference_data.get(i).getIndex() == index)
        {
          return reference_data.get(i).getType();
        }
      }
    }
    return "";
  }

  // This function prints all the reference frames if exist.
  public String printAllReferenceFrames()
  {
    String output = "";

    if (!reference_data.isEmpty())
    {
      int iteration = 0;
      for (int i=0; i<reference_data.size(); i++)
      {
        //System.out.println("output: "+output);
        if (reference_data.get(i).getIteration() != iteration)
          output += "\n"+reference_data.get(i).getIteration()+") ";
        output += reference_data.get(i).getType()+"="+reference_data.get(i).getValue()+"
";
        iteration = reference_data.get(i).getIteration();
      }
      output += "\n\n";
    }
    return "ReferenceFrame.printAllReferenceFrames(): all reference frames\n"+output;
  }
```

```
  public int getReferenceFrameSize()
  {
    if (reference_data == null)
      return 0;
    else
      return reference_data.size();
  }

  public void addReferenceData(ReferenceData w)
  {
    if (w != null)
      reference_data.add(w);
  }

  /** this function is added for Carrier */
  public boolean addReferenceData(String type, double value, int source, int frequency,
int iteration, int index)
  {
    if (type != null && frequency >0 && iteration > 0 && index >= 0)
    {
      ReferenceData rd = new ReferenceData(type, value, source, frequency, iteration,
index);
      reference_data.add(rd);
      return true;
    }
    return false;
  }

  /** this function is added for Carrier's engine, getAugmentedProfile()*/
  public void updateReferenceData(ReferenceData w, double value)
  {
    if (!reference_data.isEmpty())
    {
      for (int i=0; i<reference_data.size(); i++)
      {
        if (reference_data.get(i).getType().equals(w.getType()) &&
reference_data.get(i).getIteration() == w.getIteration() &&
reference_data.get(i).getIndex()== w.getIndex())
        {
          reference_data.get(i).updateValue(reference_data.get(i).getValue()+value);
        }
      }
    }
  }
```

```
public Vector<ReferenceData> getAllReferenceFrames()
{
  Vector<ReferenceData> allWeights;
  ReferenceData w;

  allWeights = new Vector<ReferenceData>();

  if (!reference_data.isEmpty())
  {
    for (int i=0; i<reference_data.size(); i++)
    {
      w = new ReferenceData(reference_data.get(i));
      allWeights.add(w);
    }
  }
  return allWeights;
}

/**
 * This function searches all the context weights in the reference frame
 * returns the first weight that is found match.
 */
public ReferenceData search(String type)
{
  if (!reference_data.isEmpty())
  {
    for (int i=0; i<reference_data.size(); i++)
    {
      if (reference_data.get(i).getType().equals(type))
        return reference_data.get(i);
    }
  }
  return null;
}

public int getLastIteration()
{
  int iteration = 0;
  if (!reference_data.isEmpty())
  {
    return reference_data.get(reference_data.size()-1).getIteration();
    /*
    for (int i=0; i<weights.size(); i++)
    {
      if (iteration < weights.get(i).getIteration())
```

```
            iteration = weights.get(i).getIteration();
        }
        */
    }
    return iteration;
}

// If Vector is 2 dimentional structure, then this function returns the
// column number of the structure.
public int getIndexOfLastIteration()
{
    int index = 0;
    if (!reference_data.isEmpty())
        index = reference_data.get(reference_data.size()-1).getIndex();
    return index;
}

public int getLastIndexOfIteration(int iteration)
{
    if (!reference_data.isEmpty())
    {
        int index = -1;
        for (int i=0; i<reference_data.size(); i++)
        {
            if (reference_data.get(i).getIteration() == iteration)
                index++;
            if (reference_data.get(i).getIteration() > iteration)
                break;
        }
        return index;
    }
    return 0;
}

public ReferenceFrame getOneReferenceFrame(int iteration)
{
    if (iteration < 1)
        return null;

    ReferenceFrame rf = new ReferenceFrame();

    ReferenceData rd;
    if (!reference_data.isEmpty())
    {
        for (int i=0; i<reference_data.size(); i++)
        {
          if (reference_data.get(i).getIteration() == iteration)
            rf.addReferenceData(new ReferenceData(reference_data.get(i)));
        }
      }
      return rf;
    }

    public double getMaxWeightInIteration(int iteration)
    {
      double weight = 0;
      if (!reference_data.isEmpty())
      {
        for (int i=0; i<reference_data.size(); i++)
        {
          if (reference_data.get(i).getIteration() == iteration)
            if (weight < reference_data.get(i).getValue())
              weight = reference_data.get(i).getValue();
        }
      }
      return weight;
    }

    /**
     * Insert the user generated query weight into the reference frame.
     * The reference frame is sorted in desc order.
     */
    public void insertDataInIteration(int iteration, ReferenceData w)
    {
      if (!reference_data.isEmpty())
      {
        int i = 0;
        // Find the position to insert
        for (i=0; i<reference_data.size(); i++)
        {
          if (reference_data.get(i).getIteration() == iteration)
          {
            if (reference_data.get(i).getValue() < w.getValue())
            {
              // Update the index of the newly inserted weight to the order.
              w.updateIndex(i);
              reference_data.insertElementAt(w, i);
              break;
            }
          }
        }

        // Update the rest index of the weights in the same iteration
        for (i=i+1; i<reference_data.size(); i++)
        {
          if (reference_data.get(i).getIteration() == iteration)
          {
            reference_data.get(i).updateIndex(i);
          }
        }
      }
    }

    /*
     * This function is written specially for QueryManager.
     */
    public String getReferenceData(int iteration, String location, boolean
feed_yahoo_server)
    {
      String query = "", delimiter, refined_location;
      boolean location_flag = false;

      if (feed_yahoo_server)
         delimiter = "%20";
      else
         delimiter = " ";
      refined_location = location.replaceAll(" ", delimiter);

      if (!reference_data.isEmpty())
      {
        for (int i=0; i<reference_data.size(); i++)
        {
          if (reference_data.get(i).getIteration() == iteration)
          {
             if (reference_data.get(i).getType().equals("x") ||
reference_data.get(i).getType().equals("y"))
             {
                // assume any x or y reference data points to somewhere in Miami.
                //if (query.indexOf("Miami")<0)
                //    query += delimiter+"Miami";
                location_flag = true;
             }
             else if (reference_data.get(i).getType().equals("store") ||
reference_data.get(i).getType().equals("location"))
             {
                if (query.isEmpty())
                   query = refined_location;
                else
                   query += delimiter + refined_location;
             }
             else if (!reference_data.get(i).getType().equals("speed") &&
!reference_data.get(i).getType().equals("temperature") &&
!reference_data.get(i).getType().equals("noise"))
             {
                if (query.isEmpty())
                   query = reference_data.get(i).getType();
                else
                   query += delimiter+reference_data.get(i).getType();
             }
          }
       }

       if (location_flag)
       {
         if (query.indexOf("Miami")<0)
            query += delimiter+"Miami";
       }

     }
     return query;
   }
}
/**
 *
 * Request is defined to store client request expressed in XML tag.
 * <request>search</request><value>apple</value>
 */

/**
 *
 * @author feng
 */
public class Request
{
   protected String tag_name, tag_value;
   public Request()
   {
      tag_name = "";
      tag_value = "";
   }
```

```
    public String getTagName()
    {
        return tag_name;
    }

    public String getTagValue()
    {
        return tag_value;
    }

    public void setTagName(String name)
    {
        tag_name = name;
    }

    public void setTagValue(String value)
    {
        tag_value = value;
    }
}
/*
 * Search manager provides function that uses Yahoo API to solve the
 * user queries.
 * User's selection on result documents will trigger search manager to
 * update the database. It also reorders the documents.
 */

/**
 *
 * @author feng
 */
import java.util.Vector;
import java.sql.*;
import java.util.Calendar;
import java.util.TimeZone;
import javax.swing.JOptionPane;

public class SearchManager
{
   // YahooAPI uses yahoo search API to get query results for
   // either direct query search (user query not merged with user reference
   // frame), or context search (user qeury augmented with user reference
   // frame).
   private YahooAPI yahoo;
   // QueryManager merges user query with user reference frame.
   private QueryManager query_manager;

   // document_set stores the unsorted search documents from Yahoo API
   // sorted_document_set has the sorted search documetns from Yahoo API.
   // The algorithm is based on the context profile.
   private Vector<SearchResultDocument> document_set, sorted_document_set;

   // this variable should contain all the terms from user reference frame and
   // user query.
   private String augmented_profile;

   private int DIRECT_SEARCH=1, CONTEXT_SEARCH=2,
RETURN_DOCUMENT_NUMBERS = 20;


   // MySQL database class.
   private MySQL mysql;
   private String user_query_phrase;

   public SearchManager()
   {
     yahoo = new YahooAPI();
     // Reference frames are loaded in QueryManager constructor.
     query_manager = new QueryManager();
     document_set = new Vector<SearchResultDocument>();
     sorted_document_set = new Vector<SearchResultDocument>();
   }

    /*
     * Direct query search (user query not merged with user reference frame.
     */
    public void getSearchResults(String query)
    {
       user_query_phrase = query;
       yahoo.getSearchResults(query, RETURN_DOCUMENT_NUMBERS,
document_set);
    }

    /*
     * Context query search (user query merged with user reference frame.
     */
    public void getContextSearchResults(String query)
    {
       user_query_phrase = query;
       String context_query = "";

       // if feed_yahoo_server is set to true, then query string is set to
       // feed the Yahoo search API
       boolean feed_yahoo_server;
       feed_yahoo_server = true;

       // pass user query to query_manager
       query_manager.setUserQuery(query);

       // merge user query with user reference frame.
       query_manager.setAugmentedProfile();
       context_query = query_manager.getContextQuery(feed_yahoo_server);
       //System.out.println("SearchManager.java->getContextSearchResults():
context_query = "+context_query);

       augmented_profile = query_manager.getContextQuery(false).trim();
       //System.out.println("SearchManager.java->getContextSearchResults():
augmented_profile = "+augmented_profile);

       yahoo.getSearchResults(context_query, RETURN_DOCUMENT_NUMBERS,
document_set);
    }


    // Must add "documents:" to the head of the search result.
    // Because the client site wait for this signal to switch.
    // document_set will be set either by getSearchResults() or
    // getContextSearchResults().
    public String getSearchResultString()
    {
       String result = "";

       if (document_set != null)
       {
         for (int d=0; d<document_set.size(); d++)
         {
             if (d < document_set.size()-1)
                 result += d+","+document_set.get(d).getTitle()+"|||";
             else
                 result += d+","+document_set.get(d).getTitle();
         }
       }
       if (result.length()>0)
```

```
            return "documents:"+result;
        return result;
    }


    /*
     * This function returns all document titles compiled in a string.
     * The delimiter is "|||".
     *
     */
    public String getSortedSearchResultString()
    {
        String result = "";

        if (sorted_document_set != null)
        {
            for (int d=0; d<sorted_document_set.size(); d++)
            {
                if (d < sorted_document_set.size()-1)
                    result += d+","+sorted_document_set.get(d).getTitle()+"|||";
                else
                    result += d+","+sorted_document_set.get(d).getTitle();
            }
        }
        if (result.length()>0)
            return "documents:"+result;
        return result;
    }

    /*
     * Remember to add "Summary:" to allow client to catch.
     */
    public String getSearchDocumentSummary(int document_id, int search_type)
    {
        Vector<SearchResultDocument> temp_set = new
Vector<SearchResultDocument>();
        String title="", summary="", url="";

        if (search_type == DIRECT_SEARCH)
            temp_set = document_set;
        else if (search_type == CONTEXT_SEARCH)
            temp_set = sorted_document_set;

        if (temp_set != null)
        {
          for (int d=0; d<temp_set.size(); d++)
          {
             if (Integer.parseInt(temp_set.get(d).getDocumentID()) == document_id)
             {
                title = temp_set.get(d).getTitle();
                summary = temp_set.get(d).getSummary();
                url = temp_set.get(d).getUrl();
                break;
             }
          }
          updateUserFeedBack(title, summary, url);
       }

       return "Summary: "+summary;
    }

    /*
     * update database table .
     */
    public void updateUserFeedBack(String title, String summary, String url)
    {
      mysql = new MySQL();
      Connection connection;
      PreparedStatement ps;

      boolean found = false;

      try
      {
        connection = mysql.getConnection();

        ps = connection.prepareStatement("select * from user_feed_back where
search_phrase = '" +user_query_phrase+ "' and title = '"+title+"'");
        ResultSet rs = ps.executeQuery();

        // Have to call rs.next() first.
        if (rs.next())
        {
          if (rs.getInt("id") > 0)
             found = true;
        }

        /*
        Calendar calendar = Calendar.getInstance(TimeZone.getDefault());
        statement = mysql.getConnectionStatement();
        if (!found)
          sql = "insert into user_feed_back (search_phrase, title, summary, area, url, count,
dt_modified) values ('"+
              user_query_phrase+"', '"+title+"', '"+summary+"', '33174', '"+url+"', 1, '"+
              // Note that calendar.get(calendar.MONTH) is in the range 0 to 11.
              calendar.get(Calendar.YEAR)+"-"+(calendar.get(calendar.MONTH)+1)+"-
"+calendar.get(calendar.DATE)+"
"+calendar.get(calendar.HOUR)+":"+calendar.get(calendar.MINUTE)+":"+calendar.get(
calendar.SECOND)+"')";
        else
          sql = "update user_feed_back set count = count+1, dt_modified =
'"+calendar.get(Calendar.YEAR)+"-"+(calendar.get(calendar.MONTH)+1)+"-
"+calendar.get(calendar.DATE)+"
"+calendar.get(calendar.HOUR)+":"+calendar.get(calendar.MINUTE)+":"+calendar.get(
calendar.SECOND)+"' where search_phrase = '"+user_query_phrase+"'";
        statement.executeUpdate(sql);
        statement.close();
        */

        if (!found)
        {
          // id is auto_increment field.
          // user_feed_back (search_phrase, title, summary, area, url, count, dt_modified)
          ps = connection.prepareStatement("insert into user_feed_back (search_phrase,
title, summary, area, url, count, dt_modified) values (?, ?, ?, ?, ?, ?, ?)");
          ps.setString(1, user_query_phrase);
          ps.setString(2, title);
          ps.setString(3, summary);
          ps.setString(4, "Miami");
          ps.setString(5, url);
          ps.setInt(6, 1);
          ps.setTimestamp(7, new Timestamp(System.currentTimeMillis()));
        }
        else
        {
          Calendar calendar = Calendar.getInstance(TimeZone.getDefault());
          ps = connection.prepareStatement("update user_feed_back set count = count+1,
dt_modified = '"+calendar.get(Calendar.YEAR)+"-
"+(calendar.get(calendar.MONTH)+1)+"-"+calendar.get(calendar.DATE)+"
"+calendar.get(calendar.HOUR)+":"+calendar.get(calendar.MINUTE)+":"+calendar.get(
calendar.SECOND)+"' where search_phrase = '"+user_query_phrase+"'");
        }
        ps.executeUpdate();
        connection.close();
      }
```

```
    catch(ClassNotFoundException e)
    {
      JOptionPane.showMessageDialog(null, "Mysql driver class not found.", "Alert",
JOptionPane.ERROR_MESSAGE);
    }
    catch(SQLException e)
    {
      JOptionPane.showMessageDialog(null, e.getMessage(), "Alert",
JOptionPane.ERROR_MESSAGE);
      //e.printStackTrace();
    }
  }


  /*
   *  This function evaluates context score for each search document returned
   *  from Yahoo API.
   *  The score is calculated on the basis of
   *  1) how much related the search document is to the context profile.
   *  2) other users' search history stored in user_feed_back
   */
  public void orderSearchResult()
  {
    SearchResultDocument temp_doc;
    sorted_document_set = new Vector<SearchResultDocument>();

    mysql = new MySQL();
    PreparedStatement ps;
    Connection connection;

    String summary = "";

    String[] context_profile = augmented_profile.split(" "), words;

    int d, w, s, score;

    boolean inserted = false;

    for (d=0; d<document_set.size(); d++)
    {
      score = 0;

      // 1) relativeness to user context profile
      temp_doc = new SearchResultDocument(document_set.get(d));
      summary = temp_doc.getSummary();
```

```
        for (w=0; w<context_profile.length; w++)
        {
           words = summary.split(context_profile[w]);
           score += words.length;
           //System.out.println(context_profile[w]+", score = "+score);
        }

        // 2) users' search history
        try
        {
          connection = mysql.getConnection();
          //ps = connection.prepareStatement("select count from user_feed_back where
search_phrase = '" +user_query_phrase+ "' and title = '"+temp_doc.getTitle()+"' and url =
'"+temp_doc.getUrl()+"'");
          ps = connection.prepareStatement("select count from user_feed_back where
search_phrase = ? and title = ? and url = ?");
          ps.setString(1, user_query_phrase);
          ps.setString(2, temp_doc.getTitle());
          ps.setString(3, temp_doc.getUrl());
          ResultSet rs = ps.executeQuery();
          if (rs.next())
             score += rs.getInt("count");
        }
        catch(ClassNotFoundException e)
        {
           JOptionPane.showMessageDialog(null, "Mysql driver class not found.",
"Alert", JOptionPane.ERROR_MESSAGE);
        }
        catch(SQLException e)
        {
           JOptionPane.showMessageDialog(null, e.getMessage(), "Alert",
JOptionPane.ERROR_MESSAGE);
           //e.printStackTrace();
        }

        // Reset the score for the document.
        temp_doc.setContextScore(score);

        if (sorted_document_set.size() <= 0)
           sorted_document_set.add(temp_doc);
        else
        {
           inserted = false;
           for (s=0; s<sorted_document_set.size(); s++)
           {
```

```
                    if (score > sorted_document_set.get(s).getContextScore())
                    {
                        sorted_document_set.insertElementAt(temp_doc, s);
                        inserted = true;
                        break;
                    }
                }
                if (!inserted)
                {
                    sorted_document_set.add(temp_doc);
                }
            }
        }

        // Reset document id.
        for (s=0; s<sorted_document_set.size(); s++)
        {
            sorted_document_set.get(s).setDocumentID(Integer.toString(s));
        }
    }
}
/*
 * This is the foundation class to store the document information
 * in the cache. The documents are returned from the Yahoo search API.
 */

public class SearchResultDocument
{
    private String document_id, title, summary, url;
    private int context_score;

    public SearchResultDocument()
    {
        document_id = "";
        title = "";
        summary = "";
        url = "";
        context_score = 0;
    }

    public SearchResultDocument(String title, String summary, String url, String
document_id)
    {
        this.title = title;
        this.summary = summary;
        this.url = url;
        this.document_id = document_id;
        context_score = 0;
    }

    public SearchResultDocument(SearchResultDocument srd)
    {
        title = srd.getTitle();
        summary = srd.getSummary();
        url = srd.getUrl();
        document_id = srd.getDocumentID();
        context_score = srd.getContextScore();
    }

    public String getDocumentID()
    {
        return document_id;
    }

    public String getTitle()
    {
        return title;
    }

    public String getSummary()
    {
        return summary;
    }

    public String getUrl()
    {
        return url;
    }

    public int getContextScore()
    {
        return context_score;
    }

    public void setDocumentID(String id)
    {
        document_id = id;
    }

    public void setContextScore(int score)
```

```
  {
    context_score = score;
  }
}
/*
 * Yahoo search API provides.
 * Search result document sets are in XML format.
 */

import java.net.URL;
import java.io.*;
import javax.xml.parsers.DocumentBuilderFactory;
import javax.xml.xpath.*;
import org.w3c.dom.*;
import java.util.Vector;

public class YahooAPI
{
  /**
   * This example illustrates how easy it is to parse a Yahoo! Web Service XML result
with XPath.
   */
  String Yahoo_Search_Service;
  int returned_document_number;

  public YahooAPI()
  {
    //Yahoo_Search_Service =
"http://search.yahooapis.com/WebSearchService/V1/webSearch?appid=YahooDemo&";
    Yahoo_Search_Service =
"http://api.search.yahoo.com/WebSearchService/V1/webSearch?appid=YahooDemo&";
    returned_document_number = 20;
  }

  public void getSearchResults(String query, int document_number,
Vector<SearchResultDocument> document_set)
  {
    String request;
    SearchResultDocument document;

    if (document_number == 0)
      request =
Yahoo_Search_Service+"query="+query+"&results="+returned_document_number;
    else
```

```
            request =
Yahoo_Search_Service+"query="+query+"&results="+document_number;

        try
        {
            System.out.println("YahooAPI(): "+request);
            URL url = new URL(request);
            String title, summary, urls;
            InputStream is = url.openStream();

            // Process response
            Document response =
DocumentBuilderFactory.newInstance().newDocumentBuilder().parse(is);

            XPathFactory factory = XPathFactory.newInstance();
            XPath xPath=factory.newXPath();

            // Get all search Result nodes
            // Here is the trick to skip the <ResultSet ...><Result>
            NodeList nodes = (NodeList)xPath.evaluate("/ResultSet/Result", response,
XPathConstants.NODESET);
            int nodeCount = nodes.getLength();

            //iterate over search Result nodes
            for (int i = 0; i < nodeCount; i++)
            {
                //Get each xpath expression as a string
                title = (String)xPath.evaluate("Title", nodes.item(i), XPathConstants.STRING);
                summary = (String)xPath.evaluate("Summary", nodes.item(i),
XPathConstants.STRING);
                urls = (String)xPath.evaluate("Url", nodes.item(i), XPathConstants.STRING);

                //print out the Title, Summary, and URL for each search result
                //System.out.println("Title: " + title);
                //System.out.println("Summary: " + summary);
                //System.out.println("URL: " + urls);
                //System.out.println("-----------------------------------------");

                document = new SearchResultDocument(title, summary, urls,
Integer.toString(i));
                document_set.add(document);
            }
        }
        catch (Exception e)
        {
            System.out.println("Web services request failed");
         }
      }
   }
/*
 * Data.java
 *
 * Created on January 31, 2007, 12:12 AM
 *
 * This is the base class which has the minimun members;
 * 1) type
 * 2) iteration (for ANN)
 * 3) source (from user or environment)
 */
public class Data
{
   private String type;
   private int iteration;
   private int source;

   /** Creates a new instance of Data */
   public Data(String type, int source, int iteration)
   {
      this.type = type;
      // source value:
      // 1: system generated input data
      // 2: user input data
      this.source = source;
      this.iteration = iteration;
   }

   public String getType()
   {
      return type;
   }

   public int getSource()
   {
      return source;
   }

   public int getIteration()
   {
      return iteration;
   }
```

```
}
/*
 * This is the foundation class that supports the weight
 * calculation for the reference frame and virtual frame.
 */
public class ClientData
{
  private String type;
  private double value;
  private int source;
  private int frequency;
  private int iteration;
  private int index;

  /** Creates a new instance of ClientData */
  public ClientData(String type, double value, int source, int frequency, int iteration, int
index)
  {
    this.type = type;
    this.value = value;
    // source value:
    // 1: system generated input data
    // 2: user input data
    this.source = source;
    this.frequency = frequency;
    this.iteration = iteration;
    this.index = index;
  }

  // Copy constructor
  public ClientData(ClientData cd)
  {
    this.type = cd.type;
    this.value = cd.value;
    this.frequency = cd.frequency;
    this.iteration = cd.iteration;
    this.index = cd.index;
  }

  public String getType()
  {
    return type;
  }

  public int getSource()
  {
    return source;
  }

  public double getValue()
  {
    return value;
  }

  public void updateIndex(int index)
  {
    this.index = index;
  }

  public void updateValue(double value)
  {
    this.value = value;
  }

  public int getFrequency()
  {
    return frequency;
  }

  public int getIteration()
  {
    return iteration;
  }

  public int getIndex()
  {
    return index;
  }
}
```

Appendix D. Code for Vendor Side Server Interface

```
/*
 * Author: Feng Gui
 *
 * To change this template, choose Tools | Template Manager
 * and open the template in the editor.
 */

/* Vendor.java
Setting the Main Class and Runtime Arguments
1) Right-click the project node in the Projects window and choose Project Properties.
2) Select the Run node in the Categories pane of the dialog box.
3) Type the fully qualified name of the main class in the Main Class field (for example,
   org.myCompany.myLib.MyLibClass). The main class must exist in the project or in
one
   of the JAR files or libraries on the project's runtime classpath.
4) If you use the Browse button to choose the project main class, the file chooser only
   displays classes in your project source directory. If you want to specify a class in
   one the libraries on the classpath, you have to type the fully-qualified name of the
   class in the Main Class field.
5) Enter any runtime arguments you require in the Arguments field.
   The IDE sets the project's main class and stores any newly-added arguments.
*/

import java.sql.*;
import javax.swing.*;

public class Vendor
{
    // Constructor cannot return anything.
    public Vendor()
    {
      setMysql();
    }

    public Vendor(String login, String password)
    {
       setMysql();
       if (!setVendor(login, password))
          System.out.println("Vendor(): Check login or password");
    }

    // Copy contructor
    public Vendor(Vendor vendor)
    {
```

```
  setMysql();
  setID(vendor.getID());
  setLogin(vendor.getLogin());
  setPassword(vendor.getPassword());
  setVendorName(vendor.getVendorName());
  setContact(vendor.getContact());
  setPhone(vendor.getPhone());
  setEmail(vendor.getEmail());
  setAddress1(vendor.getAddress1());
  setAddress2(vendor.getAddress2());
  setCity(vendor.getCity());
  setZip(vendor.getZip());
  setState(vendor.getState());
  setCountry(vendor.getCountry());
}

public void setMysql()
{
   mysqlDriver = "com.mysql.jdbc.Driver";
   mysqlConnectionString = "jdbc:mysql://localhost:3306/vendor";
   mysqlUser = "feng";
   mysqlPassword = "fenggui";
}

public int getID()
{
   return id;
}

public String getLogin()
{
   return login;
}

public String getPassword()
{
   return password;
}

public String getVendorName()
{
   return vendor_name;
}

public String getContact()
```

```
{
   return contact;
}

public String getPhone()
{
   return phone;
}

public String getEmail()
{
   return email;
}

public String getAddress1()
{
   return address1;
}

public String getAddress2()
{
   return address2;
}

public String getCity()
{
   return city;
}

public String getZip()
{
   return zip;
}

public int getState()
{
   return state;
}

public int getCountry()
{
   return country;
}

public void mysqlClass() throws SQLException, ClassNotFoundException
```

```
  {
     Class.forName(mysqlDriver);
     mysqlConnection = DriverManager.getConnection(mysqlConnectionString,
mysqlUser, mysqlPassword);
  }

  public void setID(int id)
  {
     this.id = id;
  }

  public void setLogin(String login)
  {
    this.login = login.trim();
  }

  public void setPassword(String password)
  {
    this.password = password.trim();
  }

  public void setVendorName(String name)
  {
     this.vendor_name = name.trim();
  }

  public void setContact(String contact)
  {
     this.contact = contact.trim();
  }

  public void setPhone(String phone)
  {
     this.phone = phone.trim();
  }

  public void setEmail(String email)
  {
     this.email = email.trim();
  }

  public void setAddress1(String address1)
  {
     this.address1 = address1.trim();
  }
```

```
public void setAddress2(String address2)
{
   this.address2 = address2.trim();
}

public void setAddress(String address1, String address2)
{
   this.address1 = address1.trim();
   this.address2 = address2.trim();
}

public void setCity(String city)
{
   this.city = city.trim();
}

public void setZip(String zip)
{
   this.zip = zip.trim();
}

public void setState(int state)
{
   this.state = state;
}

public void setCountry(int country)
{
   this.country = country;
}

 public boolean setVendor(String login, String password)
{
   String sql = "";
   boolean IsVendor = false;

   if (login != null && password!= null)
   {
      setLogin(login);
      setPassword(password);

      try
      {
         mysqlClass();
```

```
            statement = mysqlConnection.createStatement();
            sql = "select * from vendor where login = '" + login + "' and password = '" +
password + "'";
            ResultSet rs = statement.executeQuery(sql);

            // Have to call rs.next() first.
            rs.next();
            if (rs.getInt("id") > 0)
            {
               // Retrieve vendor information for later.
               setID(rs.getInt("id"));
               setVendorName(rs.getString("vendor_name"));
               setContact(rs.getString("contact"));
               setEmail(rs.getString("email"));
               setPhone(rs.getString("phone"));
               setAddress1(rs.getString("address1"));
               setAddress2(rs.getString("address2"));
               setCity(rs.getString("city"));
               setZip(rs.getString("zip"));
               setState(rs.getInt("state"));
               setCountry(rs.getInt("country"));
               IsVendor = true;
            }
            mysqlConnection.close();
         }
         catch(ClassNotFoundException e)
         {
           JOptionPane.showMessageDialog(null, e.getMessage(), "Alert",
JOptionPane.ERROR_MESSAGE);
         }
         catch(SQLException e)
         {
            JOptionPane.showMessageDialog(null, e.getMessage(), "Alert",
JOptionPane.ERROR_MESSAGE);
            //e.printStackTrace();
         }
      }
      if (IsVendor)
         return true;
      return false;
    }

   public boolean updateVendor()
   {
      boolean done = false;
```

```
        String sql;

        sql = "update vendor set vendor_name='"+getVendorName()+"',
contact='"+getContact()+"', phone='"+getPhone()+"', email='"+getEmail()+
            "', address1='"+getAddress1()+"', address2='"+getAddress2()+"',
city='"+getCity()+"', zip='"+getZip()+"', state='"+getState()+"',
country='"+getCountry()+"'"+
            "where id = "+getID();
        try
        {
          mysqlClass();
          statement = mysqlConnection.createStatement();
          statement.executeUpdate(sql);
          done = true;
        }
        catch(ClassNotFoundException e)
        {
          JOptionPane.showMessageDialog(null, e.getMessage(), "Alert",
JOptionPane.ERROR_MESSAGE);
        }
        catch(SQLException e)
        {
          JOptionPane.showMessageDialog(null, e.getMessage(), "Alert",
JOptionPane.ERROR_MESSAGE);
          //e.printStackTrace();
        }

        if (done)
          return true;
        return false;
    }

    public boolean updateVendor(String sql)
    {
        boolean done = false;
        if (sql != null || sql.length() > 0)
        try
        {
          mysqlClass();
          statement = mysqlConnection.createStatement();
          statement.executeUpdate(sql);
          done = setVendor(login, password);
        }
        catch(ClassNotFoundException e)
        {
          JOptionPane.showMessageDialog(null, e.getMessage(), "Alert",
JOptionPane.ERROR_MESSAGE);
       }
       catch(SQLException e)
       {
          JOptionPane.showMessageDialog(null, e.getMessage(), "Alert",
JOptionPane.ERROR_MESSAGE);
          //e.printStackTrace();
       }
       if (done)
          return true;
       return false;
    }

    // Mysql Login Information.
    private String mysqlDriver, mysqlConnectionString, mysqlUser, mysqlPassword;

    private Connection mysqlConnection;
    private Statement statement = null;

    private String login, password, vendor_name, contact, phone, email, address1,
address2, city, zip;
    private int id, state, country;
}
/*
 * Login.java
 *
*/
/*
Setting the Main Class and Runtime Arguments
1) Right-click the project node in the Projects window and choose Project Properties.
2) Select the Run node in the Categories pane of the dialog box.
3) Type the fully qualified name of the main class in the Main Class field (for example,
   org.myCompany.myLib.MyLibClass). The main class must exist in the project or in
one
   of the JAR files or libraries on the project's runtime classpath.
4) If you use the Browse button to choose the project main class, the file chooser only
   displays classes in your project source directory. If you want to specify a class in
   one the libraries on the classpath, you have to type the fully-qualified name of the
   class in the Main Class field.
5) Enter any runtime arguments you require in the Arguments field.
   The IDE sets the project's main class and stores any newly-added arguments.
*/

import javax.swing.*;
import java.sql.*;
import java.lang.*;

public class Login extends javax.swing.JFrame
{
    /** Creates new form Login */
    public Login()
    {
       mysqlDriver = "com.mysql.jdbc.Driver";
       mysqlConnectionString = "jdbc:mysql://localhost:3306/vendor";
       mysqlUser = "feng";
       mysqlPassword = "fenggui";
       initComponents();
       jButtonConnect.setEnabled(false);
       // Initialize for the while loop in main().
       IsVendor = false;
    }

    /** This method is called from within the constructor to
     * initialize the form.
     * WARNING: Do NOT modify this code. The content of this method is
     * always regenerated by the Form Editor.
     */
    // <editor-fold defaultstate="collapsed" desc=" Generated Code ">//GEN-
BEGIN:initComponents
    private void initComponents() {
       jPanel1 = new javax.swing.JPanel();
       jLabelLogin = new javax.swing.JLabel();
       jLabelPassword = new javax.swing.JLabel();
       jTextFieldLogin = new javax.swing.JTextField();
       jButtonConnect = new javax.swing.JButton();
       jPasswordField = new javax.swing.JPasswordField();

       setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);
       jPanel1.setBorder(javax.swing.BorderFactory.createTitledBorder("Vendor Login"));
       jLabelLogin.setText("Login");

       jLabelPassword.setText("Password");

       jTextFieldLogin.addKeyListener(new java.awt.event.KeyAdapter() {
          public void keyTyped(java.awt.event.KeyEvent evt) {
             jTextFieldLoginKeyTyped(evt);
          }
       });
```

```
        jButtonConnect.setText("Connect");
        jButtonConnect.addMouseListener(new java.awt.event.MouseAdapter() {
            public void mouseClicked(java.awt.event.MouseEvent evt) {
                jButtonConnectMouseClicked(evt);
            }
        });

        jPasswordField.addKeyListener(new java.awt.event.KeyAdapter() {
            public void keyTyped(java.awt.event.KeyEvent evt) {
                jPasswordFieldKeyTyped(evt);
            }
        });

        org.jdesktop.layout.GroupLayout jPanel1Layout = new
org.jdesktop.layout.GroupLayout(jPanel1);
        jPanel1.setLayout(jPanel1Layout);
        jPanel1Layout.setHorizontalGroup(
            jPanel1Layout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADING)
            .add(jPanel1Layout.createSequentialGroup()
                .addContainerGap()

.add(jPanel1Layout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADING)
                    .add(org.jdesktop.layout.GroupLayout.TRAILING, jLabelLogin)
                    .add(org.jdesktop.layout.GroupLayout.TRAILING, jLabelPassword))
                .addPreferredGap(org.jdesktop.layout.LayoutStyle.RELATED)

.add(jPanel1Layout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADING,
false)
                    .add(jButtonConnect)
                    .add(jTextFieldLogin, org.jdesktop.layout.GroupLayout.DEFAULT_SIZE,
159, Short.MAX_VALUE)
                    .add(jPasswordField))
                .addContainerGap(18, Short.MAX_VALUE))
        );
        jPanel1Layout.setVerticalGroup(
            jPanel1Layout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADING)
            .add(jPanel1Layout.createSequentialGroup()
                .add(19, 19, 19)

.add(jPanel1Layout.createParallelGroup(org.jdesktop.layout.GroupLayout.BASELINE)
                    .add(jLabelLogin)
                    .add(jTextFieldLogin,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE))
```

```
            .add(20, 20, 20)

.add(jPanel1Layout.createParallelGroup(org.jdesktop.layout.GroupLayout.BASELINE)
              .add(jLabelPassword)
              .add(jPasswordField, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE))
            .add(18, 18, 18)
            .add(jButtonConnect)
            .addContainerGap(17, Short.MAX_VALUE))
      );

      org.jdesktop.layout.GroupLayout layout = new
org.jdesktop.layout.GroupLayout(getContentPane());
      getContentPane().setLayout(layout);
      layout.setHorizontalGroup(
         layout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADING)
         .add(jPanel1, org.jdesktop.layout.GroupLayout.DEFAULT_SIZE,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
      );
      layout.setVerticalGroup(
         layout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADING)
         .add(jPanel1, org.jdesktop.layout.GroupLayout.DEFAULT_SIZE,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
      );
      pack();
   }// </editor-fold>//GEN-END:initComponents

   private void jPasswordFieldKeyTyped(java.awt.event.KeyEvent evt) {//GEN-
FIRST:event_jPasswordFieldKeyTyped
      // TODO add your handling code here:
      jButtonConnect.setEnabled(true);
   }//GEN-LAST:event_jPasswordFieldKeyTyped

   private void jTextFieldLoginKeyTyped(java.awt.event.KeyEvent evt) {//GEN-
FIRST:event_jTextFieldLoginKeyTyped
      // TODO add your handling code here:
      jButtonConnect.setEnabled(true);
   }//GEN-LAST:event_jTextFieldLoginKeyTyped

   private void jButtonConnectMouseClicked(java.awt.event.MouseEvent evt) {//GEN-
FIRST:event_jButtonConnectMouseClicked
      // TODO add your handling code here:
      String sql = "";
```

```
    if (jTextFieldLogin.getText() != null && jPasswordField.getPassword() != null)
    {
        this.login = jTextFieldLogin.getText();
        this.password = jPasswordField.getPassword().toString();
        this.password = "aa";
        try
        {
          mysqlClass();
          statement = mysqlConnection.createStatement();
          sql = "select * from vendor where login = '" + this.login + "' and password = '"
+ this.password + "'";
          ResultSet rs = statement.executeQuery(sql);
          System.out.println("jButtonConnectMouseClicked():"+sql);
          // Have to call rs.next() first.
          rs.next();
          if (rs.getInt("id") > 0)
          {
             IsVendor = true;
          }
          mysqlConnection.close();
        }
        catch(ClassNotFoundException e)
        {
          JOptionPane.showMessageDialog(null, e.getMessage(), "Alert",
JOptionPane.ERROR_MESSAGE);
        }
        catch(SQLException e)
        {
          JOptionPane.showMessageDialog(null, e.getMessage(), "Alert",
JOptionPane.ERROR_MESSAGE);
          //e.printStackTrace();
        }

    }

  }//GEN-LAST:event_jButtonConnectMouseClicked

  /**
   * @param args the command line arguments
   */
  /*
  public static void main(String args[])
  {
    java.awt.EventQueue.invokeLater(new Runnable() {
        public void run() {
          new Login().setVisible(true);
        }
    });
  }
  */

  public void mysqlClass() throws SQLException, ClassNotFoundException
  {
    Class.forName(mysqlDriver);
    mysqlConnection = DriverManager.getConnection(mysqlConnectionString,
mysqlUser, mysqlPassword);
  }

  public static void main(String args[])
  {
    Login login = new Login();

    while (!login.IsVendor)
    {
    // Waiting for authenticating vendor ...
    login.setVisible(true);
    }


    Vendor vendor = new Vendor(login.login, login.password);

    System.out.println("After while loop.");

    Thread campaignThread = new Thread(new campaign(vendor));
    campaignThread.setDaemon(false);
    campaignThread.start();

    System.out.println("After creating campaign.");

    login.setVisible(false);
    // 0 means normal exit.
    //System.exit(0);a
  }

  // Mysql Login Information.
  private String mysqlDriver, mysqlConnectionString, mysqlUser, mysqlPassword;
  private Connection mysqlConnection;
  private Statement statement = null;

  // Vendor information.
  private String login = "", password = "";
  private boolean IsVendor;

  // Variables declaration - do not modify//GEN-BEGIN:variables
  private javax.swing.JButton jButtonConnect;
  private javax.swing.JLabel jLabelLogin;
  private javax.swing.JLabel jLabelPassword;
  private javax.swing.JPanel jPanel1;
  private javax.swing.JPasswordField jPasswordField;
  private javax.swing.JTextField jTextFieldLogin;
  // End of variables declaration//GEN-END:variables

}
/*
 * MyItem.java
 *
* To change this template, choose Tools | Template Manager
 * and open the template in the editor.
 */
public class MyItem
{

  /** Creates a new instance of MyItem */
  public MyItem(String text, int value)
  {
    this.text = text;
    this.value = value;
  }

  public String getText()
  {
    return text;
  }

  public int getValue()
  {
    return value;
  }

  // Overrides method toString() in class Object
  // When ListModel is called, this will display text in JList.
  public String toString()
  {
    return text;
  }
    private String text = "";
    private int value;
}
/*
 * campaign.java
 *
*/
import java.util.*;
import javax.swing.*;
import java.sql.*;
import java.awt.*;
import java.lang.*;

public class campaign extends javax.swing.JFrame implements Runnable
{

    /** Creates new form campaign */
    public campaign(Vendor vendor)
    {
        initComponents();
        this.vendor = new Vendor(vendor);
        driver = "com.mysql.jdbc.Driver";
        connectionString = "jdbc:mysql://localhost:3306/vendor";
        sqlUser = "feng";
        sqlPassword = "fenggui";
        listState = new DefaultListModel();
        listClass = new DefaultListModel();
        jTabbedPane1.setTitleAt(0, "New Campaigne");
    }

    //frame.setIconImage(new ImageIcon(imgURL).getImage());



    /** This method is called from within the constructor to
     * initialize the form.
     * WARNING: Do NOT modify this code. The content of this method is
     * always regenerated by the Form Editor.
     */
    // <editor-fold defaultstate="collapsed" desc=" Generated Code ">//GEN-
BEGIN:initComponents
    private void initComponents() {
        jPanel1 = new javax.swing.JPanel();
        jLabel1 = new javax.swing.JLabel();
```

```
jTextField1 = new javax.swing.JTextField();
jLabel2 = new javax.swing.JLabel();
jTextField2 = new javax.swing.JTextField();
jButton1 = new javax.swing.JButton();
jLabel3 = new javax.swing.JLabel();
jTextField3 = new javax.swing.JTextField();
jLabel4 = new javax.swing.JLabel();
jScrollPane1 = new javax.swing.JScrollPane();
jList1 = new javax.swing.JList();
jLabel5 = new javax.swing.JLabel();
jLabel6 = new javax.swing.JLabel();
jTextField4 = new javax.swing.JTextField();
jLabel7 = new javax.swing.JLabel();
jTextField5 = new javax.swing.JTextField();
jTextField6 = new javax.swing.JTextField();
jLabel8 = new javax.swing.JLabel();
jTextField7 = new javax.swing.JTextField();
jButton2 = new javax.swing.JButton();
jTabbedPane1 = new javax.swing.JTabbedPane();
jPanel2 = new javax.swing.JPanel();
jLabelCampaignTitle = new javax.swing.JLabel();
jTextFieldCampaignTitle = new javax.swing.JTextField();
jLabelCampaignDesc = new javax.swing.JLabel();
jScrollPane2 = new javax.swing.JScrollPane();
jTextAreaDesc = new javax.swing.JTextArea();
jLabelCampaignArea = new javax.swing.JLabel();
jLabelP1L = new javax.swing.JLabel();
jLabelP2L = new javax.swing.JLabel();
jLabelP3L = new javax.swing.JLabel();
jLabelP4L = new javax.swing.JLabel();
jTextFieldX1 = new javax.swing.JTextField();
jTextFieldX2 = new javax.swing.JTextField();
jTextFieldX3 = new javax.swing.JTextField();
jTextFieldX4 = new javax.swing.JTextField();
jLabelP1C = new javax.swing.JLabel();
jLabelP2C = new javax.swing.JLabel();
jLabelP3C = new javax.swing.JLabel();
jLabelP4C = new javax.swing.JLabel();
jTextFieldY1 = new javax.swing.JTextField();
jTextFieldY2 = new javax.swing.JTextField();
jTextFieldY3 = new javax.swing.JTextField();
jTextFieldY4 = new javax.swing.JTextField();
jLabelP1R = new javax.swing.JLabel();
jLabelP2R = new javax.swing.JLabel();
jLabelP3R = new javax.swing.JLabel();
```

```
        jLabelP4R = new javax.swing.JLabel();
        jLabelClass = new javax.swing.JLabel();
        jButtonSubmit = new javax.swing.JButton();
        jScrollPane3 = new javax.swing.JScrollPane();
        jListClass = new javax.swing.JList();

        setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);
        setTitle("Vendor Campaign");
        jPanel1.setBorder(javax.swing.BorderFactory.createTitledBorder("Vendor
Information"));
        jPanel1.setInheritsPopupMenu(true);
        jPanel1.setName("");
        jLabel1.setText("Company");

        jTextField1.setFont(new java.awt.Font("Times New Roman", 0, 11));
        jTextField1.setText("jTextField1");

        jLabel2.setText("Contact");

        jTextField2.setFont(new java.awt.Font("Times New Roman", 0, 11));
        jTextField2.setText("jTextField2");

        jButton1.setText("update");
        jButton1.addMouseListener(new java.awt.event.MouseAdapter() {
            public void mouseClicked(java.awt.event.MouseEvent evt) {
                jButton1MouseClicked(evt);
            }
        });

        jLabel3.setText("Phone");

        jTextField3.setFont(new java.awt.Font("Times New Roman", 0, 11));
        jTextField3.setText("jTextField3");

        jLabel4.setText("Email");

        jList1.setFont(new java.awt.Font("Times New Roman", 0, 11));
        jList1.setModel(new javax.swing.AbstractListModel() {
            String[] strings = { "Item 1", "Item 2", "Item 3", "Item 4", "Item 5" };
            public int getSize() { return strings.length; }
            public Object getElementAt(int i) { return strings[i]; }
        });
        jList1.setSelectionMode(javax.swing.ListSelectionModel.SINGLE_SELECTION);
        jScrollPane1.setViewportView(jList1);
        jLabel5.setText("Address");

        jLabel6.setText("City");

        jTextField4.setFont(new java.awt.Font("Times New Roman", 0, 11));
        jTextField4.setText("jTextField4");

        jLabel7.setText("Zip");

        jTextField5.setFont(new java.awt.Font("Times New Roman", 0, 11));
        jTextField5.setText("jTextField5");

        jTextField6.setFont(new java.awt.Font("Times New Roman", 0, 11));
        jTextField6.setText("jTextField6");

        jLabel8.setText("State");

        jTextField7.setFont(new java.awt.Font("Times New Roman", 0, 11));
        jTextField7.setText("jTextField7");

        jButton2.setText("reset");
        jButton2.setToolTipText("Get the current vendor information");
        jButton2.addMouseListener(new java.awt.event.MouseAdapter() {
            public void mouseClicked(java.awt.event.MouseEvent evt) {
                jButton2MouseClicked(evt);
            }
        });

        org.jdesktop.layout.GroupLayout jPanel1Layout = new
org.jdesktop.layout.GroupLayout(jPanel1);
        jPanel1.setLayout(jPanel1Layout);
        jPanel1Layout.setHorizontalGroup(
            jPanel1Layout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADING)
            .add(jPanel1Layout.createSequentialGroup()
                .addContainerGap()

.add(jPanel1Layout.createParallelGroup(org.jdesktop.layout.GroupLayout.TRAILING,
false)
                    .add(jPanel1Layout.createSequentialGroup()

.add(jPanel1Layout.createParallelGroup(org.jdesktop.layout.GroupLayout.TRAILING)
                            .add(jLabel3)
                            .add(jLabel1)
                            .add(jLabel5))
                        .add(4, 4, 4)
.add(jPanel1Layout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADING,
false)
                    .add(jTextField5, org.jdesktop.layout.GroupLayout.DEFAULT_SIZE,
293, Short.MAX_VALUE)
                    .add(jTextField1, org.jdesktop.layout.GroupLayout.DEFAULT_SIZE,
293, Short.MAX_VALUE)
                    .add(jTextField3, org.jdesktop.layout.GroupLayout.DEFAULT_SIZE,
293, Short.MAX_VALUE)))
              .add(org.jdesktop.layout.GroupLayout.LEADING,
jPanel1Layout.createSequentialGroup()
                  .add(31, 31, 31)
                  .add(jLabel7)
                  .addPreferredGap(org.jdesktop.layout.LayoutStyle.RELATED)
                  .add(jTextField7, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE,
0, Short.MAX_VALUE)))
            .add(29, 29, 29)

.add(jPanel1Layout.createParallelGroup(org.jdesktop.layout.GroupLayout.TRAILING)
              .add(jLabel2)
              .add(jLabel4)
              .add(jLabel8)
              .add(jLabel6))
            .addPreferredGap(org.jdesktop.layout.LayoutStyle.RELATED)

.add(jPanel1Layout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADING,
false)

.add(jPanel1Layout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADING,
false)
                  .add(jTextField6, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE,
274, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE)
                  .add(jTextField4, org.jdesktop.layout.GroupLayout.DEFAULT_SIZE,
274, Short.MAX_VALUE)
                  .add(jTextField2, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE,
274, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE)
                  .add(org.jdesktop.layout.GroupLayout.TRAILING,
jPanel1Layout.createSequentialGroup()
                    .add(jButton1)
                    .add(16, 16, 16)
                    .add(jButton2)))
              .add(jScrollPane1, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE,
146, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE))
            .add(37, 37, 37))
    );
```

```
        jPanel1Layout.linkSize(new java.awt.Component[] {jButton1, jButton2},
org.jdesktop.layout.GroupLayout.HORIZONTAL);

        jPanel1Layout.linkSize(new java.awt.Component[] {jTextField1, jTextField2,
jTextField3, jTextField4, jTextField5, jTextField6, jTextField7},
org.jdesktop.layout.GroupLayout.HORIZONTAL);

        jPanel1Layout.setVerticalGroup(
            jPanel1Layout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADING)
            .add(jPanel1Layout.createSequentialGroup()

.add(jPanel1Layout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADING)
                    .add(jPanel1Layout.createSequentialGroup()
                        .add(34, 34, 34)

.add(jPanel1Layout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADING)
                            .add(jPanel1Layout.createSequentialGroup()

.add(jPanel1Layout.createParallelGroup(org.jdesktop.layout.GroupLayout.BASELINE)
                                    .add(jLabel1)
                                    .add(jTextField1,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE))
                                .addPreferredGap(org.jdesktop.layout.LayoutStyle.RELATED)

.add(jPanel1Layout.createParallelGroup(org.jdesktop.layout.GroupLayout.BASELINE)
                                    .add(jLabel3)
                                    .add(jTextField3,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE, 18,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE))
                                .add(6, 6, 6)

.add(jPanel1Layout.createParallelGroup(org.jdesktop.layout.GroupLayout.BASELINE)
                                    .add(jLabel5)
                                    .add(jTextField5,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE)))
                            .add(jPanel1Layout.createSequentialGroup()
                                .add(72, 72, 72)

.add(jPanel1Layout.createParallelGroup(org.jdesktop.layout.GroupLayout.BASELINE)
                                    .add(jLabel7)
```

```
                              .add(jTextField7,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE)))))
                    .add(jPanel1Layout.createSequentialGroup()
                        .add(31, 31, 31)

.add(jPanel1Layout.createParallelGroup(org.jdesktop.layout.GroupLayout.BASELINE)
                            .add(jLabel2)
                            .add(jTextField2,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE, 18,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE))
                        .addPreferredGap(org.jdesktop.layout.LayoutStyle.RELATED)

.add(jPanel1Layout.createParallelGroup(org.jdesktop.layout.GroupLayout.BASELINE)
                            .add(jLabel4)
                            .add(jTextField4,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE))
                        .addPreferredGap(org.jdesktop.layout.LayoutStyle.RELATED)

.add(jPanel1Layout.createParallelGroup(org.jdesktop.layout.GroupLayout.BASELINE)
                            .add(jTextField6,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE)
                            .add(jLabel6))
                        .addPreferredGap(org.jdesktop.layout.LayoutStyle.RELATED)

.add(jPanel1Layout.createParallelGroup(org.jdesktop.layout.GroupLayout.TRAILING)
                            .add(jLabel8)
                            .add(jScrollPane1,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE, 19,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE))))
                .addPreferredGap(org.jdesktop.layout.LayoutStyle.RELATED, 28,
Short.MAX_VALUE)

.add(jPanel1Layout.createParallelGroup(org.jdesktop.layout.GroupLayout.BASELINE)
                    .add(jButton2)
                    .add(jButton1))
                .addContainerGap())
        );
```

```
    jPanel2.setBorder(javax.swing.BorderFactory.createTitledBorder("Campaign
Detail"));
    jLabelCampaignTitle.setText("Title");

    jLabelCampaignDesc.setText("Description");

    jTextAreaDesc.setColumns(20);
    jTextAreaDesc.setRows(5);
    jScrollPane2.setViewportView(jTextAreaDesc);

    jLabelCampaignArea.setText("Campaign Area");

    jLabelP1L.setText("(");

    jLabelP2L.setText("(");

    jLabelP3L.setText("(");

    jLabelP4L.setText("(");

    jTextFieldX1.setFont(new java.awt.Font("Times New Roman", 0, 11));
    jTextFieldX1.setText("X1");
    jTextFieldX1.addFocusListener(new java.awt.event.FocusAdapter() {
      public void focusGained(java.awt.event.FocusEvent evt) {
        jTextFieldX1FocusGained(evt);
      }
      public void focusLost(java.awt.event.FocusEvent evt) {
        jTextFieldX1FocusLost(evt);
      }
    });
    jTextFieldX1.addMouseListener(new java.awt.event.MouseAdapter() {
      public void mouseClicked(java.awt.event.MouseEvent evt) {
        jTextFieldX1MouseClicked(evt);
      }
    });

    jTextFieldX2.setFont(new java.awt.Font("Times New Roman", 0, 11));
    jTextFieldX2.setText("X2");

    jTextFieldX3.setFont(new java.awt.Font("Times New Roman", 0, 11));
    jTextFieldX3.setText("X3");

    jTextFieldX4.setFont(new java.awt.Font("Times New Roman", 0, 11));
    jTextFieldX4.setText("X4");
jLabelP1C.setText(",");

jLabelP2C.setText(",");

jLabelP3C.setText(",");

jLabelP4C.setText(",");

jTextFieldY1.setFont(new java.awt.Font("Times New Roman", 0, 11));
jTextFieldY1.setText("Y1");

jTextFieldY2.setFont(new java.awt.Font("Times New Roman", 0, 11));
jTextFieldY2.setText("Y2");

jTextFieldY3.setFont(new java.awt.Font("Times New Roman", 0, 11));
jTextFieldY3.setText("Y3");

jTextFieldY4.setFont(new java.awt.Font("Times New Roman", 0, 11));
jTextFieldY4.setText("Y4");

jLabelP1R.setText(")");

jLabelP2R.setText(")");

jLabelP3R.setText(")");

jLabelP4R.setText(")");

jLabelClass.setText("Class");

jButtonSubmit.setText("Submit");
jButtonSubmit.addMouseListener(new java.awt.event.MouseAdapter() {
   public void mouseClicked(java.awt.event.MouseEvent evt) {
      jButtonSubmitMouseClicked(evt);
   }
});

jListClass.setFont(new java.awt.Font("Times New Roman", 0, 11));
jListClass.setModel(new javax.swing.AbstractListModel() {
   String[] strings = { "Item 1", "Item 2", "Item 3", "Item 4", "Item 5" };
   public int getSize() { return strings.length; }
   public Object getElementAt(int i) { return strings[i]; }
});
jScrollPane3.setViewportView(jListClass);
```

```
        org.jdesktop.layout.GroupLayout jPanel2Layout = new
org.jdesktop.layout.GroupLayout(jPanel2);
        jPanel2.setLayout(jPanel2Layout);
        jPanel2Layout.setHorizontalGroup(
            jPanel2Layout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADING)
            .add(jPanel2Layout.createSequentialGroup()
                .addContainerGap()

.add(jPanel2Layout.createParallelGroup(org.jdesktop.layout.GroupLayout.TRAILING)
                    .add(jButtonSubmit)
                    .add(org.jdesktop.layout.GroupLayout.LEADING,
jPanel2Layout.createSequentialGroup()

.add(jPanel2Layout.createParallelGroup(org.jdesktop.layout.GroupLayout.TRAILING)
                            .add(jLabelCampaignDesc)
                            .add(jLabelCampaignTitle))
                        .addPreferredGap(org.jdesktop.layout.LayoutStyle.RELATED)

.add(jPanel2Layout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADING,
false)
                            .add(jTextFieldCampaignTitle)
                            .add(jScrollPane2,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE, 270,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE))
                        .add(15, 15, 15)

.add(jPanel2Layout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADING)
                            .add(org.jdesktop.layout.GroupLayout.TRAILING, jLabelClass)
                            .add(org.jdesktop.layout.GroupLayout.TRAILING,
jLabelCampaignArea))
                        .addPreferredGap(org.jdesktop.layout.LayoutStyle.RELATED)

.add(jPanel2Layout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADING)
                            .add(jPanel2Layout.createSequentialGroup()

.add(jPanel2Layout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADING,
false)
                                    .add(jLabelP3L,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                                    .add(jLabelP1L))
                                .add(4, 4, 4)

.add(jPanel2Layout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADING,
false)
```

```
                                .add(jPanel2Layout.createSequentialGroup()
                                    .add(jTextFieldX1,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE, 40,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE)
                                    .addPreferredGap(org.jdesktop.layout.LayoutStyle.RELATED)
                                    .add(jLabelP1C))
                                .add(jPanel2Layout.createSequentialGroup()
                                    .add(jTextFieldX3,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE, 40,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE)
                                    .addPreferredGap(org.jdesktop.layout.LayoutStyle.RELATED)
                                    .add(jLabelP3C,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)))
                            .addPreferredGap(org.jdesktop.layout.LayoutStyle.RELATED)

.add(jPanel2Layout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADING,
false)
                                .add(jPanel2Layout.createSequentialGroup()
                                    .add(jTextFieldY1,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE, 40,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE)
                                    .addPreferredGap(org.jdesktop.layout.LayoutStyle.RELATED)
                                    .add(jLabelP1R))
                                .add(jPanel2Layout.createSequentialGroup()
                                    .add(jTextFieldY3,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE, 40,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE)
                                    .addPreferredGap(org.jdesktop.layout.LayoutStyle.RELATED)
                                    .add(jLabelP3R,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)))
                            .add(20, 20, 20)

.add(jPanel2Layout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADING,
false)
                                .add(jLabelP4L,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                                .add(jLabelP2L))
                            .add(4, 4, 4)

.add(jPanel2Layout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADING)
                                .add(jPanel2Layout.createSequentialGroup()
                                    .add(jTextFieldX4,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE, 40,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE)
                                    .addPreferredGap(org.jdesktop.layout.LayoutStyle.RELATED)
                                    .add(jLabelP4C)
                                    .addPreferredGap(org.jdesktop.layout.LayoutStyle.RELATED)
                                    .add(jTextFieldY4,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE, 40,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE)
                                    .addPreferredGap(org.jdesktop.layout.LayoutStyle.RELATED)
                                    .add(jLabelP4R))
                                .add(jPanel2Layout.createSequentialGroup()
                                    .add(jTextFieldX2,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE, 40,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE)
                                    .addPreferredGap(org.jdesktop.layout.LayoutStyle.RELATED)
                                    .add(jLabelP2C)
                                    .addPreferredGap(org.jdesktop.layout.LayoutStyle.RELATED)
                                    .add(jTextFieldY2,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE, 40,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE)
                                    .addPreferredGap(org.jdesktop.layout.LayoutStyle.RELATED)
                                    .add(jLabelP2R))))
                            .add(jScrollPane3, org.jdesktop.layout.GroupLayout.DEFAULT_SIZE,
236, Short.MAX_VALUE))))
                .addContainerGap(62, Short.MAX_VALUE))
        );
        jPanel2Layout.setVerticalGroup(
            jPanel2Layout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADING)
            .add(jPanel2Layout.createSequentialGroup()
                .addContainerGap()

.add(jPanel2Layout.createParallelGroup(org.jdesktop.layout.GroupLayout.BASELINE)
                    .add(jLabelCampaignTitle)
                    .add(jTextFieldCampaignTitle,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE))
                .add(16, 16, 16)

.add(jPanel2Layout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADING)
                    .add(jPanel2Layout.createSequentialGroup()

.add(jPanel2Layout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADING)
                            .add(jPanel2Layout.createSequentialGroup()
.add(jPanel2Layout.createParallelGroup(org.jdesktop.layout.GroupLayout.BASELINE)
                              .add(jLabelP2L)
                              .add(jLabelP2C)
                              .add(jTextFieldX2,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE)
                              .add(jTextFieldY2,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE)
                              .add(jLabelP2R))
                           .add(35, 35, 35)

.add(jPanel2Layout.createParallelGroup(org.jdesktop.layout.GroupLayout.BASELINE)
                              .add(jLabelP4L)
                              .add(jTextFieldX4,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE)
                              .add(jLabelP4C)
                              .add(jTextFieldY4,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE)
                              .add(jLabelP4R)))
                        .add(jPanel2Layout.createSequentialGroup()

.add(jPanel2Layout.createParallelGroup(org.jdesktop.layout.GroupLayout.BASELINE)
                              .add(jLabelP1L)
                              .add(jLabelP1C)
                              .add(jTextFieldY1,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE)
                              .add(jLabelP1R)
                              .add(jTextFieldX1,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE)
                              .add(jLabelCampaignArea))
                           .add(35, 35, 35)

.add(jPanel2Layout.createParallelGroup(org.jdesktop.layout.GroupLayout.BASELINE)
                                .add(jLabelP3L)
                                .add(jTextFieldX3,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE)
                                .add(jLabelP3C)
                                .add(jTextFieldY3,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE)
                                .add(jLabelP3R))))
                        .add(29, 29, 29)

.add(jPanel2Layout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADING)
                            .add(jScrollPane3, org.jdesktop.layout.GroupLayout.DEFAULT_SIZE,
100, Short.MAX_VALUE)
                            .add(jLabelClass)))
                    .add(jLabelCampaignDesc)
                    .add(jScrollPane2, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE,
200, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE))
                .add(29, 29, 29)
                .add(jButtonSubmit)
                .add(20, 20, 20))
        );
        jTabbedPane1.addTab("tab1", jPanel2);

        org.jdesktop.layout.GroupLayout layout = new
org.jdesktop.layout.GroupLayout(getContentPane());
        getContentPane().setLayout(layout);
        layout.setHorizontalGroup(
            layout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADING)
            .add(jPanel1, org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, 750,
Short.MAX_VALUE)
            .add(jTabbedPane1, org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, 750,
Short.MAX_VALUE)
        );
        layout.setVerticalGroup(
            layout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADING)
            .add(layout.createSequentialGroup()
                .add(jPanel1, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE)
                .add(17, 17, 17)
                .add(jTabbedPane1, org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, 376,
Short.MAX_VALUE))
```

```
        );
        pack();
    }// </editor-fold>//GEN-END:initComponents

    private void jTextFieldX1FocusLost(java.awt.event.FocusEvent evt) {//GEN-
FIRST:event_jTextFieldX1FocusLost
        // TODO add your handling code here:
        if (jTextFieldX1.getText().length()==0)
        {
          jTextFieldX1.setText("X1");
        }
    }//GEN-LAST:event_jTextFieldX1FocusLost

    private void jTextFieldX1FocusGained(java.awt.event.FocusEvent evt) {//GEN-
FIRST:event_jTextFieldX1FocusGained
        // TODO add your handling code here:
        if (jTextFieldX1.getText().equals("X1"))
        {
          jTextFieldX1.setText("");
        }
    }//GEN-LAST:event_jTextFieldX1FocusGained

    private void jTextFieldX1MouseClicked(java.awt.event.MouseEvent evt) {//GEN-
FIRST:event_jTextFieldX1MouseClicked
        // TODO add your handling code here:
    }//GEN-LAST:event_jTextFieldX1MouseClicked

    private void jButtonSubmitMouseClicked(java.awt.event.MouseEvent evt) {//GEN-
FIRST:event_jButtonSubmitMouseClicked
       // TODO add your handling code here:
       boolean update = false;
       if (!checkField(jTextFieldCampaignTitle, "campaign title"))
       {
         //JOptionPane.showMessageDialog(null, "Vendor Name Error", "Error",
JOptionPane.ERROR_MESSAGE);
         jTextFieldCampaignTitle.setBackground(new Color(255,255,0));
         jTextFieldCampaignTitle.requestFocusInWindow();
         return;
       }
       else if (!checkField(jTextAreaDesc, "description"))
       {
         //JOptionPane.showMessageDialog(null, "Contact Error", "Error",
JOptionPane.ERROR_MESSAGE);
         jTextAreaDesc.setBackground(new Color(255,255,0));
         jTextAreaDesc.requestFocusInWindow();
```

```
          return;
        }
    }//GEN-LAST:event_jButtonSubmitMouseClicked

    private void jButton2MouseClicked(java.awt.event.MouseEvent evt) {//GEN-
FIRST:event_jButton2MouseClicked
        // TODO add your handling code here:
        showVendorInformation(vendor);
        jTextField1.setBackground(new Color(255,255,255));
        jTextField2.setBackground(new Color(255,255,255));
        jTextField3.setBackground(new Color(255,255,255));
        jTextField4.setBackground(new Color(255,255,255));
        jTextField5.setBackground(new Color(255,255,255));
        jTextField6.setBackground(new Color(255,255,255));
        jTextField7.setBackground(new Color(255,255,255));
    }//GEN-LAST:event_jButton2MouseClicked

    private void jButton1MouseClicked(java.awt.event.MouseEvent evt) {//GEN-
FIRST:event_jButton1MouseClicked
        // TODO add your handling code here:

        boolean update = false;
        if (!checkField(jTextField1, "vendor name"))
        {
          //JOptionPane.showMessageDialog(null, "Vendor Name Error", "Error",
JOptionPane.ERROR_MESSAGE);
          jTextField1.setBackground(new Color(255,255,0));
          jTextField1.requestFocusInWindow();
          return;
        }
        else if (!checkField(jTextField2, "contact"))
        {
          //JOptionPane.showMessageDialog(null, "Contact Error", "Error",
JOptionPane.ERROR_MESSAGE);
          jTextField2.setBackground(new Color(255,255,0));
          jTextField2.requestFocusInWindow();
          return;
        }
        else if (!checkField(jTextField3, "phone"))
        {
          //JOptionPane.showMessageDialog(null, "Phone Error", "Error",
JOptionPane.ERROR_MESSAGE);
          jTextField3.setBackground(new Color(255,255,0));
          jTextField3.requestFocusInWindow();
          return;
```

```
      }
      else if (!checkField(jTextField4, "email"))
      {
        //JOptionPane.showMessageDialog(null, "Email Error", "Error",
JOptionPane.ERROR_MESSAGE);
        jTextField4.setBackground(new Color(255,255,0));
        jTextField4.requestFocusInWindow();
        return;
      }
      else if (!checkField(jTextField5, "address"))
      {
        //JOptionPane.showMessageDialog(null, "Address Error", "Error",
JOptionPane.ERROR_MESSAGE);
        jTextField5.setBackground(new Color(255,255,0));
        jTextField5.requestFocusInWindow();
        return;
      }
      else if (!checkField(jTextField6, "city"))
      {
        //JOptionPane.showMessageDialog(null, "City Error", "Error",
JOptionPane.ERROR_MESSAGE);
        jTextField6.setBackground(new Color(255,255,0));
        jTextField6.requestFocusInWindow();
        return;
      }
      else if (!checkField(jTextField7, "zip"))
      {
        //JOptionPane.showMessageDialog(null, "Zip Error", "Error",
JOptionPane.ERROR_MESSAGE);
        jTextField7.setBackground(new Color(255,255,0));
        jTextField7.requestFocusInWindow();
        return;
      }
      else if (jList1.getSelectedIndex() < 0)
      {
        JOptionPane.showMessageDialog(null, "Must select state", "Error",
JOptionPane.ERROR_MESSAGE);
        return;
      }

      if (!jTextField1.getText().equals(vendor.getVendorName()))
      {
         vendor.setVendorName(jTextField1.getText());
         update = true;
      }
      if (!jTextField2.getText().equals(vendor.getContact()))
      {
         vendor.setContact(jTextField2.getText());
         update = true;
      }
      if (!jTextField3.getText().equals(vendor.getPhone()))
      {
         vendor.setPhone(jTextField3.getText());
         update = true;
      }
      if (!jTextField4.getText().equals(vendor.getEmail()))
      {
         vendor.setEmail(jTextField4.getText());
         update = true;
      }
      if (vendor.getAddress2() == "" &&
!jTextField5.getText().equals(vendor.getAddress1()))
      {
         vendor.setAddress1(jTextField5.getText());
         update = true;
      }
      if (!jTextField6.getText().equals(vendor.getCity()))
      {
         //update += "city ='"+jTextField6.getText().trim()+"',";
         vendor.setCity(jTextField6.getText());
         update = true;
      }

      if (!jTextField7.getText().equals(vendor.getZip()))
      {
         //update += "zip ='"+jTextField7.getText().trim()+"',";
         vendor.setZip(jTextField7.getText());
         update = true;
      }

      MyItem selectedState = (MyItem)
jList1.getModel().getElementAt(jList1.getSelectedIndex());
      if (selectedState.getValue() != vendor.getState())
      {
         vendor.setState(selectedState.getValue());
         update = true;
      }

      if (update)
      {
          if (vendor.updateVendor())
             JOptionPane.showMessageDialog(null, "Vendor information has been updated.
Reset will reflect updated information", "Note",
JOptionPane.INFORMATION_MESSAGE);
          else
             JOptionPane.showMessageDialog(null, "Vendor information update failed",
"Error", JOptionPane.ERROR_MESSAGE);
       }
       else
          JOptionPane.showMessageDialog(null, "No change in vendor information.",
"Alert", JOptionPane.INFORMATION_MESSAGE);

    }//GEN-LAST:event_jButton1MouseClicked

    private boolean checkField(JTextField tf, String type)
    {
       String string = tf.getText();
       if (string == "" || string.length()==0)
       {
          JOptionPane.showMessageDialog(null, "Please enter "+type, "Error",
JOptionPane.ERROR_MESSAGE);
          return false;
       }
       return true;
    }

    private boolean checkField(JTextArea ta, String type)
    {
       String string = ta.getText();
       if (string == "" || string.length()==0)
       {
          JOptionPane.showMessageDialog(null, "Please enter "+type, "Error",
JOptionPane.ERROR_MESSAGE);
          return false;
       }
       return true;
    }
    /**
     * @param args the command line arguments
     */
    /*
    public static void main(String args[])
    {
       java.awt.EventQueue.invokeLater
       (
          new Runnable()
          {
             public void run()
             {
                // Must create an object.
                campaign c = new campaign();
                if (!c.authenticate())
                {
                   JOptionPane.showMessageDialog(null, "Login Failed", "Error",
JOptionPane.ERROR_MESSAGE);
                   return;
                }
                c.showVendorInformation();
                c.setVisible(true);
             }
          }
       );
    }
    */
    // test campaign thread.
    // Comment out main() to run as a thread.

    public static void main(String args[])
    {
      Vendor vendor = new Vendor("aa", "aa");
      Thread t = new Thread(new campaign(vendor));
      t.start();
    }
    // test campaign thread.

    public void run()
    {
      showVendorInformation(this.vendor);
      setVisible(true);
    }

    public void showState()
    {
       try
       {
          Class.forName(driver);
          connection = DriverManager.getConnection(connectionString, sqlUser,
sqlPassword);
          statement = connection.createStatement();
          String sql = "select * from state";
```

```
            ResultSet rs = statement.executeQuery(sql);

            while (rs.next())
            {
               //listState.addElement(rs.getString("state"));
               listState.addElement(new MyItem(rs.getString("state"), rs.getInt("id")));
            }

            jList1.setModel(listState);

            // Select the state for the vendor.
            if (listState.getSize() > vendor.getState())
            {
               jList1.setSelectedIndex(vendor.getState()-1);
               jList1.ensureIndexIsVisible(vendor.getState()-1);
            }
            jList1.setSelectionMode(ListSelectionModel.SINGLE_SELECTION);
            connection.close();
         }
         catch(Exception e)
         {
            JOptionPane.showMessageDialog(null, e.getMessage(), "alert",
JOptionPane.ERROR_MESSAGE);
            //e.printStackTrace();
         }
      }

      public void showClass()
      {
         try
         {
            Class.forName(driver);
            connection = DriverManager.getConnection(connectionString, sqlUser,
sqlPassword);
            statement = connection.createStatement();
            String sql = "select * from class";
            ResultSet rs = statement.executeQuery(sql);

            while (rs.next())
            {
               listClass.addElement(new MyItem(rs.getString("class_name"), rs.getInt("id")));
            }
            jListClass.setModel(listClass);
            connection.close();
         }
```

```
        catch(Exception e)
        {
            JOptionPane.showMessageDialog(null, e.getMessage(), "alert",
JOptionPane.ERROR_MESSAGE);
            //e.printStackTrace();
        }

    }

    public void showVendorInformation(Vendor vendor)
    {
      jTextField1.setText(vendor.getVendorName());
      if (vendor.getContact() != null)
          jTextField2.setText(vendor.getContact());
      jTextField3.setText(vendor.getPhone());
      jTextField4.setText(vendor.getEmail());
      if (vendor.getAddress2() == null)
         jTextField5.setText(vendor.getAddress1());
      else
         jTextField5.setText(vendor.getAddress1()+" "+vendor.getAddress2());
      jTextField6.setText(vendor.getCity());
      jTextField7.setText(vendor.getZip());
      showState();
      showClass();
    }

    // Vendor information
    private Vendor vendor;

    // Mysql Login Information.
    private String driver, connectionString, sqlUser, sqlPassword;
    private Connection connection;
    private Statement statement = null;

    private DefaultListModel listState, listClass;

    // Variables declaration - do not modify//GEN-BEGIN:variables
    private javax.swing.JButton jButton1;
    private javax.swing.JButton jButton2;
    private javax.swing.JButton jButtonSubmit;
    private javax.swing.JLabel jLabel1;
    private javax.swing.JLabel jLabel2;
    private javax.swing.JLabel jLabel3;
    private javax.swing.JLabel jLabel4;
    private javax.swing.JLabel jLabel5;
    private javax.swing.JLabel jLabel6;
    private javax.swing.JLabel jLabel7;
    private javax.swing.JLabel jLabel8;
    private javax.swing.JLabel jLabelCampaignArea;
    private javax.swing.JLabel jLabelCampaignDesc;
    private javax.swing.JLabel jLabelCampaignTitle;
    private javax.swing.JLabel jLabelClass;
    private javax.swing.JLabel jLabelP1C;
    private javax.swing.JLabel jLabelP1L;
    private javax.swing.JLabel jLabelP1R;
    private javax.swing.JLabel jLabelP2C;
    private javax.swing.JLabel jLabelP2L;
    private javax.swing.JLabel jLabelP2R;
    private javax.swing.JLabel jLabelP3C;
    private javax.swing.JLabel jLabelP3L;
    private javax.swing.JLabel jLabelP3R;
    private javax.swing.JLabel jLabelP4C;
    private javax.swing.JLabel jLabelP4L;
    private javax.swing.JLabel jLabelP4R;
    private javax.swing.JList jList1;
    private javax.swing.JList jListClass;
    private javax.swing.JPanel jPanel1;
    private javax.swing.JPanel jPanel2;
    private javax.swing.JScrollPane jScrollPane1;
    private javax.swing.JScrollPane jScrollPane2;
    private javax.swing.JScrollPane jScrollPane3;
    private javax.swing.JTabbedPane jTabbedPane1;
    private javax.swing.JTextArea jTextAreaDesc;
    private javax.swing.JTextField jTextField1;
    private javax.swing.JTextField jTextField2;
    private javax.swing.JTextField jTextField3;
    private javax.swing.JTextField jTextField4;
    private javax.swing.JTextField jTextField5;
    private javax.swing.JTextField jTextField6;
    private javax.swing.JTextField jTextField7;
    private javax.swing.JTextField jTextFieldCampaignTitle;
    private javax.swing.JTextField jTextFieldX1;
    private javax.swing.JTextField jTextFieldX2;
    private javax.swing.JTextField jTextFieldX3;
    private javax.swing.JTextField jTextFieldX4;
    private javax.swing.JTextField jTextFieldY1;
    private javax.swing.JTextField jTextFieldY2;
    private javax.swing.JTextField jTextFieldY3;
    private javax.swing.JTextField jTextFieldY4;
    // End of variables declaration//GEN-END:variables
```

VITA

FENG GUI

1994 - 1999	Bachelor in Science in Computer Engineering University of Alberta
1999 - 2002	HCET Florida International University Miami, Florida
2001	SGI Certified IRIX System Administrator SGI Global Education
2001 - 2003	Master in Science in Computer Engineering Florida International University
2003 - 2009	Doctoral Candidate in Electrical Engineering Florida International University

PUBLICATION

F. Gui, M. Guillen, N. Rishe, A. Barreto, J. Andrian, M. Adjouadi, "A Client-Server Architecture for Context-Aware Search Application", the 12th International Conference Network-Based Information systems, August 19-21, 2009, Indianapolis.

F. Gui, M. Adjouadi, N. Rishe, "A Contextualized and Personalized Approach for Mobile Search", 2nd International Workshop on Data Management for Wireless and Pervasive Communications, May 26-29, 2009, University of Bradford, Bradford, UK

F. Gui, M. Adjouadi, N. Rishe, "Personalized Approach for Mobile Search", World Congress on Computer Science and Information Engineering, March 31 - April 2, 2009, Los Angeles/Anaheim, CA.

A. Gan, F. Gui, I. Ubaka, "A Web-Based System for Comprehensive Analysis of National Transit Database", 10th International Conference on the Application Advanced Technologies in Transportation (AATT), Athens, May, 2008

F. Gui, N. Zong M. Adjouadi, "Artificial intelligence approach of context-awareness architecture for mobile computing", 6th International Conference Intelligent Systems Design and Applications-ISDA, 7 pp. October 16-18, 2006, Los Alamitos, CA.

N. Zong, F. Gui, M. Adjouadi, “A New Clustering Algorithm of Large Datasets with O(N) Computational Complexity”, 5th International Conference Intelligent Systems Design and Applications-ISDA, pp. 79-82,Wroclaw, Poland, September 8-10, 2005.

F. Gui, N. Zong, M. Adjouadi, "Dynamic Neural Network Based Algorithm for Context Awareness in Mobile Computing", WSEAS Transactions on Communications, Vol. 4 (8), pp. 629-636, August 2005.

F. Gui, C. Liu, D. Wang, and X. Chen “A Context-Awareness Algorithm Based On Data Cache For Mobile Computing”, World Wireless Congress, WWC'2004, San Francisco, Session S05.01: Mobility, Handoff, Ad-Hoc & Networks, May 25-28, 2004.