CONTEXT BASED CONTENT MANAGEMENT FOR LOCATION-AWARE WEB APPLICATIONS

Priyank Panchal
B.E., Gujarat University, India, 2008

PROJECT

Submitted in partial satisfaction of the requirements for the degree of

MASTER OF SCIENCE

in

ELECTRICAL AND ELECTRONIC ENGINEERING

at

CALIFORNIA STATE UNIVERSITY, SACRAMENTO

SUMMER
2011
CONTEXT BASED CONTENT MANAGEMENT FOR LOCATION-AWARE WEB APPLICATIONS

A Project

by

Priyank Panchal

Approved by:

__________________________________, Committee Chair
Jing Pang, Ph.D.

____________________________, Second Reader
Suresh Vadhva, Ph.D.

____________________________
Date

ii
Student: Priyank Panchal

I certify that this student has met the requirements for format contained in the University format manual and that this project is suitable for shelving in the Library and credit is to be awarded for the Project

__________________________, Department Chair  _______________________
Suresh Vadhva, Ph.D.  Date

Department of Electrical and Electronic Engineering
Abstract

of

CONTEXT BASED CONTENT MANAGEMENT FOR LOCATION-AWARE WEB APPLICATIONS

by

Priyank Panchal

This project focuses on the context-aware system support applications that make use of context information including user location, time and date, neighboring environment etc. with the contents uploaded over internet. Such types of applications are easily integrated in mobile devices such as PDA, mobile phones and so on. In addition, multimedia contents may also be added for extended data management to be applicable over a large group of consumers.

This project describes a simple algorithm implemented on a small database. The project work has been completed in two languages: MySQL and PHP. MySQL is used to create and maintain the database while making it easily searchable by issuing related queries. PHP or Hypertext Preprocessor is the language used to create the interface between webpage and the database. When used along with MySQL, it enhanced the database searching while giving the freedom of a scripting language. As the multimedia content in the database increases, there is practically no room left for any search that
cannot return anything related to the subject matter. Thus, it becomes highly necessary to constrain the data retrieval from the database to better suit what the user is looking for.

In addition, this project provides a practical example by restricting the search based on the distance from the geographic location of the user. In this project, a simple algorithm to constrain the search from the database based on the most popular entry within a given distance is demonstrated.

__________________________, Committee Chair
Jing Pang, Ph.D.

__________________________
Date
ACKNOWLEDGEMENTS

I would like to present Dr. Jing Pang my deepest thanks for giving me the opportunity to extend my project experience into this realm. This project, with all its richness in the research areas, also provided me with the insight into how the major web applications are made. I was further enlightened with the knowledge of PHP and MySQL for data management.

I would also like to thank Dr. Preetham Kumar and Dr. Suresh Vadhva, for reviewing my project report and being supportive throughout the career of the project work.

I would also like to present appreciation towards the faculty members of Department of Electrical and Electronics Engineering at California State University, Sacramento, who provided their help, and support through their knowledge towards the completion of my graduation.

Lastly, I would like to thank my family and friends for their support both academically, as well as personally throughout my academic career.

Priyank Panchal
# TABLE OF CONTENTS

<table>
<thead>
<tr>
<th>Acknowledgements</th>
<th>vi</th>
</tr>
</thead>
<tbody>
<tr>
<td>List of Tables</td>
<td>ix</td>
</tr>
<tr>
<td>List of Figures</td>
<td>x</td>
</tr>
</tbody>
</table>

## Chapter

1. INTRODUCTION ........................................................................................................... 1
    1.1 Introduction to the Context Based Content Management System ............. 1
    1.2 Purpose of the Project ..................................................................................... 2
    1.3 Significance of the Project .............................................................................. 4
    1.4 Organization of the Report ............................................................................. 4

2. CONTEXT BASED CONTENT MANAGEMENT ................................................. 6
    2.1 Introduction to the Concept of Context ....................................................... 6
    2.2 Concept and Relationships ............................................................................ 9
    2.3 Attributed Relational Graphs ....................................................................... 11

3. UBIQUITOUS COMPUTING ................................................................................. 14
    3.1 Introduction to Ubiquitous Computing .......................................................... 14
    3.2 Application/User Mobility ............................................................................. 17
LIST OF TABLES

Table 4.1 Database table structure for system 1 ................................................................. 27
Table 4.2 Database table structure for system 2 ................................................................. 28
Table 5.1 Field variables for the project system ............................................................... 33
Table 5.2 Access control of the variables ......................................................................... 37
# LIST OF FIGURES

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.1</td>
<td>The ARG of movement of one user among different locations</td>
<td>12</td>
</tr>
<tr>
<td>3.1</td>
<td>Ubiquitous computing</td>
<td>20</td>
</tr>
<tr>
<td>4.1</td>
<td>Framework of the system</td>
<td>25</td>
</tr>
<tr>
<td>4.2</td>
<td>Popularity index ‘v’ of a node</td>
<td>30</td>
</tr>
<tr>
<td>5.1</td>
<td>Data describing fields in the database</td>
<td>34</td>
</tr>
<tr>
<td>5.2</td>
<td>Environment describing fields in the database</td>
<td>34</td>
</tr>
<tr>
<td>5.3</td>
<td>System specification with variables and users</td>
<td>38</td>
</tr>
<tr>
<td>5.4</td>
<td>Variable reaction in context</td>
<td>39</td>
</tr>
<tr>
<td>5.5</td>
<td>Search system interface architecture</td>
<td>40</td>
</tr>
<tr>
<td>5.6</td>
<td>Web search page</td>
<td>41</td>
</tr>
<tr>
<td>5.7</td>
<td>Search result for the system</td>
<td>42</td>
</tr>
<tr>
<td>5.8</td>
<td>Mapping API interface</td>
<td>43</td>
</tr>
<tr>
<td>5.9</td>
<td>Web interface for system 2</td>
<td>44</td>
</tr>
<tr>
<td>5.10</td>
<td>Input fields for database on web interface</td>
<td>45</td>
</tr>
</tbody>
</table>
Chapter 1

INTRODUCTION

This chapter discusses about context based content management systems that are so much discussed into as the management of more and more multimedia becomes cumbersome. New methods and application need to be developed to make maximum use of the data into the database as well efficiently store the newer data that is into the database.

The chapter progresses to discuss the project perspective into the context based content management systems. The latter part of the chapter focuses on the significance of the project and the progress of the report.

1.1 Introduction to the Context Based Content Management System

Nowadays, multimedia contents and mobile devices are becoming more and more popular. Images, voice and text-message, GPS functionalities on mobile devices have become daily experience for a lot of people. Using a smart phone, it is possible to gather above data and store them into a database on a web server.

Today, there are a lot of data grids present to accommodate the need for storage of large multimedia contents [1]. But, with increasing pool of multimedia contents, the problem still resides in the retrieval of contextual data from the multimedia pool. Advanced techniques are required to retrieve particular types of data as the major problem that lies in the multimedia data is psychological side of human perception. The human perception of particular multimedia content may differ from another’s perception
of the same type of content. According to L. Dunckley, “this leads to a large semantic gap between the actual data and the human perception about it.”[2] This makes it inevitable to have a management system that defines an algorithm to help in effective retrieval of data from the multimedia pool. This management system is widely known as context based content management system.

With increasing number of users, the complexity of mining out the relevant data from the database also increases [3]. Thus, it becomes inevitable to develop an algorithm that can manage relevant data from specific content such as manage location-aware data.

The approach towards developing this project is a thorough understanding of context-based management system and developing an interface to mine out data from the database that can be helpful to the user in his/her current context which in the current scenario is the location of the user.

1.2 Purpose of the Project

The purpose of the project is to develop an algorithm that implements the relational search based on the geographic location of the user. The project database is a context based database where a number of fields are included along with a multimedia entry to enhance the definition of the entry and facilitate the context based search.

This type of search is very common nowadays, as the complexity as well as the depth of the matter of multimedia content in the database is increasing everyday [3].

There are two systems developed that demonstrate two ways to mine out relevant data from the system as well recommend the next possible data that the user would
require. Since, the context is the location of the user, the database has been designed with considerations about storing location based information. A map based interface has been used to facilitate the user to understand the location.

The first system is divided into two main parts. In the first part, the keywords are asked for request from the user and are submitted to the data-mining script in the back-end of the web page. These keywords are then looked upon for matching counterparts in the database. The second part implements an algorithm that sorts the data on the basis of most popular content as well as proximity context of the user, i.e. how much nearer the searched location is to the user. Finally, the search is returned and the geographic location of the search result is returned to the Google Maps Application Programmable Interface (API). The API then marks the location of the search result. Coherently, the whole process is displayed back on the search page with the results of the search from the location, and the map showing the location of the search.

The second system utilizes the concept of rating based recommender systems. Such recommender systems are used to recommend the user a location based on the description of his context and based upon the past ratings of the different users. Thus, this approach can help provide the most relevant data to the current user according to the liking of the other previous users. This system then automatically updates the new information entered by the user and consequently changes the recommendations for future users.
1.3 Significance of the Project

Context based multimedia management system is a very rapid growing field as the quantity of multimedia content used daily by people is increasing. Analysts and researchers have long worked on developing algorithms and concepts in mining the data from the database. These techniques involve use of filter-by-form techniques where the search criteria are logically compared against the search [3]. This project works on similar grounds. The search is logically compared against the form entries and the search result is filtered out.

This project is a small-scale research on the modern techniques implemented in the most advanced and sophisticated geographic search engines. Several systems and applications can be made using these techniques and mapping them with different mapping interfaces [4]. Development of such a system allows search and mining of data relevant to the geographic location of the user, which is also one of the several contexts related to the user. The two systems discussed in the previous section can be integrated together in a single system to form an advanced “Location Recommender System” in which the current GPS data of a particular location can be stored. Based upon this GPS data, users can be recommended for a place in a particular scenario.

1.4 Organization of the Report

The project report is been divided into several parts.
Chapter 2 focuses on the study of various context-based management systems. This provides a general understanding of how context-based applications can be developed and what features are taken into account.

Chapter 3 gives an in-depth description of Ubiquitous Computing and how it can be achieved in the modern applications. It focuses on how context-based systems can be used in Ubiquitous Computing applications.

Chapter 4 discusses the design implementation for developing a location-aware application. It gives an overview behind how the system of the current project is developed.

Chapter 5 is a detailed explanation of the design implementation of the current project. It also demonstrated the functionality of the current project application.

Finally, chapter 6 concludes the project work by discussing the key features of the current project work and enlightening the vast future perspectives that the current system holds.
Chapter 2

CONTEXT BASED CONTENT MANAGEMENT

This chapter focuses on the concept of context and the relationship that can be determined between different contextual content that is stored in the database. As the depth and width of the database increases, more and more researches are made to design relationships between the different data based upon their contextual relevance. This enables the efficient retrieval of data as well as creates new horizons where new data can be recommended to users. Such systems, which utilize the relationship techniques between the data to recommend new information useful to the user, are now commonly known as recommender systems.

2.1 Introduction to the Concept of Context

It has been long since scientists and researchers have quarreled over the definition of context that shall be used to develop artificial intelligence.

The word context has specific meaning as it deals not only with the data content in the system but also with the context in which the data was used.

According to Oxford University Press, the context means “the circumstances that form the setting of an event, statement, or idea, and in terms of which it can be fully understood and assessed” [5].

According to Brown et al., context is defined as “the location, identities around the user, the time of the day, season, temperature, etc.” [6]. A similar definition is presented by Ryan et al. where context is defined as “user’s location, environment,
identity and time” [7]. While Dey in [8] explicitly defines context as “user’s emotional state, focus of attention, location and orientation, date and time, objects and people in user’s environment”.

Although there are similar synonymous definitions of context presented, the most explicit definition has been presented again by Dey et al. in [9] as:

“Context is 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.”

But this definition of the word context does not come handy when it is being used for applications in ubiquitous computing. People when communicating with one another, are able to analyze the implicit information or context and thus can have increased communication bandwidth. But, such rich contextual information is not transferred implicitly when communication takes place between people and mobile devices [9].

The important aspect is the human perception of the data, which is the context with which he/she is uploading into the server. This context can be different for different individuals.

To minimize this human perception error, there are many attempts to categorize the contexts, which lay importance to some context parameters and let the computer effectively understand the context of the user. This is done by the application designer, who can determine the reasons leading to the occurrence of certain events of the user and helps determine a near to actual mapping of the user’s context [10]. For example, if a user
is following a trail of tourist attractions, then it is with the context of visiting places of attraction in the proximity that he is looking for.

According to report [9], we can define context types for characterizing the situation of a particular context. Thus the location, time, date and activity are defined as primary context types. Defining primary context types gives away a lot of information about the user such as - phone number, etc. Such related information is categorized into the secondary context information. Thus items related to the primary context type are defined as secondary context information.

The use of secondary contextual information is to conglomerate related information. Hence, related objects in the user’s entity may share the same contextual perspective and thus such objects can be included to define common context of the user. Using such common contextual information, it is possible to retrieve new information for the user. For example, if a user has a definitive pattern in his/choice of locations according to the time of the day then such a time-dependent activity can be used to relate to his/her choice of locations. This leads to further refine his search for locations based upon time. In a simple scenario, this can be exemplified by establishing relationships between the user and his friends. Hence, by knowing the primary contextual information about a user, one can know about his friends, and relationships with other people. In addition, this leads to recommend the user relevant information based upon what people in his group are searching for.

The purpose of context based data querying is to provide relevant data mining based on the search. This can be established by the use of various fields to enhance
understanding of the context of the data stored in the database. Since, all the previous definitions of the context have various fields to describe the context of the user, there exists a need to simplify the information to concentrate upon.

According to [10], the three most important aspects of context are where the user is, who is he/she with, and what the resources available near him/her.

But, whatever input factors we consider to define the context of the application usage, these context-aware parameters are not constant with user. Thus, it is required to re-assess these parameters on a constant basis and change the application behavior whenever there is change in these parameters. Such an application, which can understand the changing environment around the user, and output the information that is usable to the user in his present context, are known as context-aware application.

The most efficient definition of a context-aware application is provided by Dey et al. in [9] as

“A system is context aware if it uses context to provide relevant information and/or services to the user, where relevancy depends upon user’s task.”

2.2 Concept and Relationships

To make development of such a self-informative application, it is necessary to have as much information about the context from the user as possible. According to [10], if a system is to be context-aware, then it is required to detect, interpret, and respond to context.
The main objective behind enhancing the querying of the multimedia content from the database is to provide a result that is most relevant in the context that the user is looking for. This is enabled by increasing the entries in the database that improve the description of the multimedia such as providing date of entry, relevant geographic location, full textual description of the multimedia content, story synopsis in the case of videos, etc.

Providing such a vast content along with the data lets the user to search from a number of fields and this increases the probability of mining the most relevant data.

This kind of database querying approach is accomplished by many modern applications such as Google Maps. The relational database is looked upon for particular entries searched by the user and also obtain data related to the search. This results in generation of a large database array consisting of elements all fulfilling the search query. But, what makes it an interesting topic is how these results are sorted in a manner relevant to the user. The search results are further sorted out based upon another set of contextual criteria, which can be either – supplied by the user, or can be specific for a specific kind of application. Some of the context criterions that are looked upon to further finesse the search results are nearest geographic locations (also known as the proximity criterion) and popularity of the search result. The data obtained from the database are looked upon to output the ones that are nearest to a specific geographic location. This enables the user to look for the search nearest to his geographic location.
Thus, to enhance the searching capability of the database, it is required to have a sophisticated database from which the querying can be easily done.

2.3 Attributed Relational Graphs

The data stored in the database effectively accumulates user-related information into a collection. But, to utilize this data to build a successful recommender system, one needs to define the relationships between activities of the user as well as one user’s activity with that of other user’s activities. This is accomplished using Attributed Relational Graph or ARG.

Attributed Relational Graph is a technique for visually analyzing real-life data. The concept is that any kind of graph can be converted by adding discrete feature vectors to the vertices [3]. Thus when applied to the real-life data graphs, with a certain attribute between the nodes added, these graphs can reveal the relationship between the nodes based upon that attribute.

For example, when a website and the links associated with it are represented as an ARG, the web pages can be the vertex and the URL links can be links resembling the vertices. The relationship attributed can be applied as the relationship between a particular webpage and the other pages and the graph can be used to analyze which page is easily accessible while which is not [4].

To understand the purpose of the project, an ARG on life-log data was studied [2]. The various life-log data consists of collection of each user’s moving path data. These
paths when used as ARGs with nodal description of the location, are frequently used for querying the similar-type entries from the multimedia database which can also be related to the particular geographic location.

One such Attributed Relational Graph is depicted in the following figure. Here, the ARG is made of a particular user as his movements among different locations. This gives the relationship between the nodes (with respect to the user), which here represent the stops he has made. The relation attribute is strongly depicted as the line and number between the nodes.

![Attributed Relational Graph](image)

**Figure 2.1** The ARG of movement of one user among different locations

Thus, by drawing the ARG of a particular entity and its related entities, we can understand the relationship of that entity with the others. This property when implemented in the context of the project, with the geographic locations taken as node, the relationship of one node with other nodes sharing a common property may lead us to design the next spot a person would like to travel from that node. Thus, if a person wants to travel from a particular node, the relationship factor pertaining to the search of a
particular type of locations, may give us the next most probable node from the current node. In this manner, we can propose a path to travel for a person.
Chapter 3

UBIQUITOUS COMPUTING

The main reason for the need to make the computer applications artificially intelligent can be attributed to as eventually making the computer application self-sufficient in every aspect of human behavior. Such an application, which can easily integrated in every need of people, or which can be made ubiquitous, are known as Ubiquitous Systems.

3.1 Introduction to Ubiquitous Computing

Many papers have already been published which describe the development of a ubiquitous computing system. But, the main question that arises while discussing the architecture of such a system is “what is meant by ubiquitous computing?”.

Ubiquitous computing is described by the authors of [14] as “a new genre of computing that completely permeates the life of the user”. They further describe the ubiquitous computing as a method by which computers do not come in the way of the users i.e. are invisible to the users, but, at the same time, assist the users by performing tasks that the user intends to do, and meeting his/her needs.

Thus, ubiquitous computing applications may refer to any type of computer applications that can perform or assist the user in his or her work without the user having to actually trigger the software. Thus, this definitively excludes any applications that are based upon traditional desktop-based. According to [15], ubiquitous computing or “ubicomp” is described as “an emerging paradigm for interaction between people and
computers”. The same authors also describe it as “a breakaway from the current paradigm of desktop computing to provide computational services to the user when and where required.”[15]

Thus, to adequately describe the ubiquitous systems, we have to first define the realms or limits to which the ubiquitous systems extend up to. According to [15], a ubiquitous system consists of 1) Heterogeneous set of computing devices 2) a set of supported tasks and 3) optional infrastructure the devices may rely upon to carry out the supported tasks. This optional infrastructure may be anything that the application may use for its support, and may be any other computing application, or a network, GPS which may provide the location data, etc.

Thus, a ubiquitous computing application does not just depend upon the interweaving or interdependence of the different computing software, but also to other hardware components or different other devices, for example a GPS, to complete the application.

But, still unclear about the exact realm that a ubiquitous application should focus upon, we need to discuss the major properties of ubiquitous computing. According to Weiser [16], there are two inherent properties of a ubiquitous computing application namely

1. Ubiquity
2. Transparency
By “ubiquity” of the computer system, we mean that the system or application should be accessible by the user whenever and wherever he or she needs it. It is inherent to the basic definition of the ubiquitous system which suggests the system to be universally available or being “ubiquitous”. Such a ubiquity has been possible by the onset of modern devices and other networks such as Mobile and Wireless networks. Thus, with respect to the modern applications, the main concern for a system to be ubiquitous is to make the system more mobile. This in turn, enables user to be remain mobile, while not being location-dependent to access the system.

Transparency is another important aspect of the Ubiquitous system, refers that the system or application should be “non-intrusive and be able to integrate into the everyday environment” [16]. Transparency is synonymously used with the invisibility of the execution of the computer application. Invisibility, further not only refers to the invisibility from the user’s eyes of the execution of the computer application, but also refers to the unconscious use of the application that aids the user’s actions.

Thus, an ideal ubiquitous system should not only be mobile in nature that it can be used anywhere and anytime, but also be able to able to aid the user without consciously being initiated by the user. These features of mobility and invisibility are further describes as follows.
3.2 Application/User Mobility

A typical computer application is bounded by a certain limit that describes the user’s movement. For example, such a limit may be in terms of spatial dimensions or temporal dimensions. Spatial dimensions, as the name suggests, reflect the limitation of the user to use the application within a certain range of distance. Such a type of limitation, as is with most of the cases, is due to the connectivity to the resources, for example, the network.

With the advent of wireless networks, it has been possible to overcome the spatial limitation for developing a ubicomp application. Wireless network; refer to the use of wireless technologies to connect the computer to another computer or a network. This has been, since a long time, attracted millions of consumers and as such, the applications have become mobile. As described by [14], the wireless networks allows the users to escape the tether of a network cable and allows the user to access computer and network services from anywhere within the reach of a wireless network.

3.3 Transparency

Application is defined to be transparent if it is able to execute without the conscious knowledge of the user. This has a direct implication on the ability of the application to be able to serve the user for the information that the user needs, without the actual triggering of the application.
Another aspect of an application being transparent is that the user should not be spending time in thinking how to run the application [14]. If the application cannot suffice this aspect, the user is generally preoccupied with the redundant duplicate task of using the computer as well extracting the information or services he/she requires [14]. A very subtle example of an actual ubiquitous system would be an automatic door that senses a person’s approach, and opens the door without causing the user to open the door whenever he confronts the entry/exit portal. Thus, the user is free from engaging himself in the obvious task of opening the door whenever he is approaching. This application makes the activity of opening a door as a redundant activity with respect to user, as the system senses that the user intends to open the door by the cause that he is approaching the door. Donald Norman, exemplifies this situation by stressing the point that he would not like a word processor where he actually has to perform tasks to tell the application that he intends to write a letter, but a letter writer which can allow him to write the letter more efficiently without being hampered by the word processing tools.

One major type of transparency that is the concern of the modern application, as pointed out by [15] is interaction transparency. The author describes that the interaction transparency “applies to the system’s interface and reflects the conscious efforts and attention the system requires of the user either for operating it or for perceiving the output”.

In modern computer application, the main hindrance that transparency is causing is the interaction transparency. According to [15], while performing any task with
modern systems, there is a need where the user is required to understand the interface. The user is also not at the liberty to perform the tasks without any hindrance, as he has to perceive the interaction and hence there is the necessity to perform the dual tasks to understand the interface. This emphasizes the necessity for a transparent interface that, is unconsciously present invisible to the user, and hence saves the user from paying any attention to the background application. This is one way of achieving interaction transparency.

Another way, as described by [17], to achieve interaction transparency is to take some tasks from the hands of the user and handling the responsibility to the system, without the active indulgence of the user. This is known as “task migration”. A simple example of task migration can be given by describing the way modern dynamic websites work. The collection of geographic location through the IP address of the computer enables the websites to restrict the search content of the user. This relieves the user to specify the geographic location when he/she is looking for a location nearby.

As pointed out by [15], the transparency and mobility of any system with the classification of being ubiquitous can be graphically described as below.
Thus, with this characterization, it is clearer that while augmented reality concentrates on providing more application and interface transparency, desktop computing is the basic form of computing that does not concentrate enough on both the forms of the ubiquitous computing.
The transparency of any ubiquitous computing has been defined further by [15] by proposing two perceptive terms of transparency – *semantic transparency* and *syntactic transparency*.

### 3.4 Semantic Transparency

Semantic Transparency has been defined as the transparency of the system for which the system can predict or anticipate what the user is intending to do. The best example for this is the automatic door example as explained in the previous section.

### 3.5 Syntactic Transparency

Syntactic Transparency is characterized in situations where the application does not deduce the intentions of the user, but on the contrary produces or introduces the tasks for the user. For example, while composing a document, the word processing document produces tasks for the user, where it prompts for saving the document at regular intervals. This helps the user to overcome the tedious task of remembering to save the document while he is engrossed in composing the document.

Another example of syntactic transparency is retrieving of emails from the server at regular intervals. Although, the task performed by the application does not reflect the actual task that the user intended to perform, it prompts the user for further actions, like reading or replying to emails.
3.6 Ubiquitous Computing and Context-Aware Applications

Since the ubiquitous systems are self-capable of performing tasks of the user without the requirement of external prompt, they are many times required to be context-aware.

As the mobility of the system in increased, the location of the user changes drastically. The task that the user is required or intended to perform often changes with the scenario or location of the user. Hence, here it is possible to describe a new definition of context-aware where an application is able to understand the environment and scenario around the user and develop the tasks that the user intends to perform. Thus, the ubiquitous systems are required to be context-aware so as to understand or adapt to the new user environment and change the tasks that the user needs to perform. One such application example can be a simple search engine that changes the search results according to the location of the user, so as to provide better location-aware searches for the user.

Thus, as the application moves towards full mobility, in the ubiquitous matrix, it is essentially required for the system to be context-aware. The context, although, predominantly refers to the location of the user, other contexts such as frequently used searches, can also be used together to enhance the system performance. Such a system is the core objective of the project and is described in the following sections.
Chapter 4
LOCATION AWARE APPLICATION

This chapter is dedicated to the discussion about the implementation perspective of a context-aware application. Since, the context most commonly utilized and understood by the computer applications is the location of the user, the application built here is a location-aware application. The chapter progresses by giving an overview of the requirements and features of a location-aware application to the actual implementation of the project.

The two systems of the project are described in detail in the latter part of the chapter.

4.1 Background of Location-Aware Applications

Today, when applications are slowly moving from desktop-based applications to mobile applications, it has got attention from many researchers to provide location-aware ubiquitous applications. Modern applications require ubiquitous in location-aware context in order to provide services to the user as he moves freely in the spatial dimensions. As, the location based context of the user changes, the user needs to continuously tell the application about difference in parameters that he is inputting.

One such application example is the location-based search for the user. As, the location of the user changes, the user needs the application to provide search results
which are local to the location. This has a direct implication for a system to point to the current location of the user and provide a context-based search.

This project is based upon context based content rated search. The system is designed where the user is able to rate a particular location that provides the search basis for future searches by the same user as well as other users.

This chapter has been divided into various sections from introduction to the framework of the design to the development of the application. Different aspects of the applications are also discussed.

4.2 Framework of the System

The development of any system is based upon a strong framework that supplies the useful characteristics of interdependencies among different structures of the system. This system has also been developed on a similar framework as explained in the figure below.
The major part of the system is the framework server which stores all information about the user preferred locations as well as the map database. This enables the user to access information from anywhere in the world wherever the user is connected to the internet network. Thus, the user can be as mobile as possible within the range of wireless networks. This provides the support for mobility which is also the primitive objective of a ubiquitous system.
Another major component of the design is the web interface that enables the users to interact with the server through their mobile device as shown in the framework. This web interface allows the users to search for the locations as well to store multiple locations on the map which can be referred later on.

The data in the server are stored in two formats namely, in the form of MySQL database, as well as a free script file in the form format in an XML file. XML stands for Extensive Markup Language, which although the name is misleading, is not a replacement for the Hyper Text Markup Language or HTML, which is commonly used to develop the front-end services for a web interface. The XML format is used to store and retrieve data and not used to display the data. The data are still displayed using the HTML format on the web interface.

The MySQL database is used to store the data, as it provides easier retrieval of the data with simplistic queries.

Another important development in the project has been the usage of a map Application Programmable Interface or API. These map API are provided by many sources available and can be freely integrated into the interface to display and make use of already built maps, for the specific application. Some of the examples of services that provide the mapping API are Google Maps™, Bing® Maps, Yahoo!® Maps, OpenLayers, Mapstraction, etc. In the current project, Google Maps API has been used for simplification, but other mapping APIs can be used and different preloaded maps are available.
4.3 Database Development

The most important framework in the current project is the development of the database. The database here is developed in MySQL, but can also be developed on other server based databases that are very common in the current market such as Oracle. MySQL is a relational database management system (RDBMS), and is commonly used to manage large chunks of data [5].

The database table field for system 1 which allows the user to search from a pre-defined database is described below.

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
<th>Null</th>
<th>Key</th>
<th>Default</th>
<th>Extra</th>
</tr>
</thead>
<tbody>
<tr>
<td>Image</td>
<td>varchar(20)</td>
<td>YES</td>
<td></td>
<td>Null</td>
<td></td>
</tr>
<tr>
<td>Fname</td>
<td>varchar(20)</td>
<td>YES</td>
<td></td>
<td>Null</td>
<td></td>
</tr>
<tr>
<td>Lname</td>
<td>varchar(20)</td>
<td>YES</td>
<td></td>
<td>Null</td>
<td></td>
</tr>
<tr>
<td>Descr</td>
<td>varchar(20)</td>
<td>YES</td>
<td></td>
<td>Null</td>
<td></td>
</tr>
<tr>
<td>Locx</td>
<td>varchar(30)</td>
<td>YES</td>
<td></td>
<td>Null</td>
<td></td>
</tr>
<tr>
<td>Locy</td>
<td>varchar(30)</td>
<td>YES</td>
<td></td>
<td>Null</td>
<td></td>
</tr>
<tr>
<td>search_hit</td>
<td>int(11)</td>
<td>NO</td>
<td>PRI</td>
<td>Null</td>
<td>auto_increment</td>
</tr>
<tr>
<td>Id</td>
<td>int(11)</td>
<td>NO</td>
<td>PRI</td>
<td>Null</td>
<td></td>
</tr>
</tbody>
</table>

Table 4.1 Database table structure for system 1

The database tables developed store context based information into the database such as location co-ordinates, user details and time when the location is entered by the user. In table 4.1, different fields are provided to accommodate the basic information of
the location. This table is used with the data pre-loaded into the database such as GPS data of any location. When the user searches for a particular location, its popularity index is improved by increasing the “search_hit” field value thus making it more recommended in similar searches by other users.

A similar table is developed for the rating based user recommender system that enables users to save the location through the interface, unlike the other system, in which the database needs to be pre-loaded. The structure of the table is shown in table 4.2

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
<th>Null</th>
<th>Key</th>
<th>Default</th>
<th>Extra</th>
</tr>
</thead>
<tbody>
<tr>
<td>Id</td>
<td>int(11)</td>
<td>NO</td>
<td>PRI</td>
<td>NULL</td>
<td>auto_increment</td>
</tr>
<tr>
<td>Name</td>
<td>varchar(50)</td>
<td>YES</td>
<td>NULL</td>
<td>NULL</td>
<td></td>
</tr>
<tr>
<td>Descry</td>
<td>varchar(100)</td>
<td>YES</td>
<td>NULL</td>
<td>NULL</td>
<td></td>
</tr>
<tr>
<td>Lat</td>
<td>varchar(20)</td>
<td>YES</td>
<td>NULL</td>
<td>NULL</td>
<td></td>
</tr>
<tr>
<td>Lng</td>
<td>varchar(20)</td>
<td>YES</td>
<td>NULL</td>
<td>NULL</td>
<td></td>
</tr>
<tr>
<td>Addr</td>
<td>varchar(100)</td>
<td>YES</td>
<td>NULL</td>
<td>NULL</td>
<td></td>
</tr>
<tr>
<td>cur_timestamp</td>
<td>timestamp</td>
<td>NO</td>
<td>CURRENT_TIMESTAMP</td>
<td>on update CURRENT_TIMESTAMP</td>
<td></td>
</tr>
<tr>
<td>User</td>
<td>varchar(50)</td>
<td>YES</td>
<td>NULL</td>
<td>NULL</td>
<td></td>
</tr>
<tr>
<td>ratingValue</td>
<td>int(3)</td>
<td>YES</td>
<td>NULL</td>
<td>NULL</td>
<td></td>
</tr>
<tr>
<td>Count</td>
<td>int(11)</td>
<td>YES</td>
<td>NULL</td>
<td>NULL</td>
<td></td>
</tr>
<tr>
<td>Average</td>
<td>float</td>
<td>YES</td>
<td>NULL</td>
<td>NULL</td>
<td></td>
</tr>
</tbody>
</table>

Table 4.2 Database table structure for system 2

In the database structure for the second system, the field "cur_timestamp" is added. This field is automatically updated whenever the user changes the entry in the field table. This is a simple and efficient way to determine when the user has been to the location in question in the past.
4.4 Search Algorithm Development

For any system to be self-sufficient, it needs to have algorithm that can enable it to provide with optimum solutions. With reference to the current project, the application should be self-sufficient to provide optimum search capabilities. Thus, the searching algorithm will be much more sophisticated. The most common algorithm is the one that displays the most popular search algorithm. But, popularity is in itself a concept that can be evaluated differently as explained in the following section.

The search for popularity is determined by converting each location as a node and building up an ARG for the same. The relationship between different nodes is determined by the edges connecting the nodes.

The simplest popularity index of a node is calculated as the sum of the nodes with links pointing to that particular node [3]. In this project, it is implemented as described in equation 4.1.

\[ a(v) = \sum_{1}^{N} w \]  

Thus, a particular node can be considered more popular than other nodes if it has been linked to more nodes than the other nodes in question. This is the basic algorithm that is developed for setting up the search algorithm for a web site where the popularity is of a page based upon how many other pages are pointing towards it.
Figure 4.2 Popularity index ‘ν’ of a node

In the current project, the popularity of a particular location as saved by the user is determined by how many users have visited it. This can also be considered as how many users have asked for the search of the particular location in question. Thus, if any location has been asked for more times than another location, it is considered to be more popular. Hence, every time when a user looks for that particular location, then the popularity index of that particular location is increased by 1.

Moreover, the user locations are rated by different users and the ratings are compared with other locations in the same category. This refines the recommendations based upon user ratings. Thus, if any location has been rated by n users, each rating it as
Then, the average rating of the location on a general realm is calculated to determine the popularity index as given in equation 4.2.

\[
\text{Average rating} = \frac{\sum_{i=0}^{n} R_i}{n}
\]  

Such kind of indexing of locations, gives us a fine approximation of the popularity of a particular location among the users. Thus, if a user has been to a particular location, then the recommendation is based upon his past activities regarding that location. This is known as personalized search. Another aspect of personalized search is if a user is searching for a location that he has not been to, then the recommendation is based upon the popularity of the location among other users which share a relation with the current user. If the user is searching anonymously, then the search results are non-personalized since they do not depend upon the past activities of the user.

As the last step to build a ubiquitous recommender system, the web interface is developed which acts as a medium to communicate between humans and the computers. This is discussed in the next chapter.
Chapter 5

DESIGN IMPLEMENTATION

This chapter is dedicated to the work performed during the course of the project. Initially, the chapter focuses on the system specification of the design. As the chapter progresses, the design implementation is discussed considering the different scenarios of the application.

5.1 System Specification

The system specification of the project reflects the organization on which the project was built. The main objective of the system was to build a location-aware context-based application. Databases discussed in the previous chapter were developed to include fields that reflect the contextual information. These databases are further discussed in more detail in the following sections.

This project tried to collect enough data about the user’s environment.

The database field variables discussed previous chapter, are collectively tabulated in Table 5.1 described on the next page.
<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
<th>Null</th>
<th>Key</th>
<th>Default</th>
<th>Extra</th>
</tr>
</thead>
<tbody>
<tr>
<td>Id</td>
<td>int(11)</td>
<td>NO</td>
<td>PRI</td>
<td>NULL</td>
<td>auto_increment</td>
</tr>
<tr>
<td>Name</td>
<td>varchar(50)</td>
<td>YES</td>
<td>NULL</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Descr</td>
<td>varchar(100)</td>
<td>YES</td>
<td>NULL</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Lat</td>
<td>varchar(20)</td>
<td>YES</td>
<td>NULL</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Lng</td>
<td>varchar(20)</td>
<td>YES</td>
<td>NULL</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Addr</td>
<td>varchar(100)</td>
<td>YES</td>
<td>NULL</td>
<td></td>
<td></td>
</tr>
<tr>
<td>cur_timestamp</td>
<td>timestamp</td>
<td>NO</td>
<td>CURRENT_TIMESTAMP</td>
<td>on update CURRENT_TIMESTAMP</td>
<td></td>
</tr>
<tr>
<td>User</td>
<td>varchar(50)</td>
<td>YES</td>
<td>NULL</td>
<td></td>
<td></td>
</tr>
<tr>
<td>tel_num</td>
<td>int(10)</td>
<td>YES</td>
<td>NULL</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Profession</td>
<td>Varchar(100)</td>
<td>YES</td>
<td>NULL</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Age</td>
<td>int(3)</td>
<td>NO</td>
<td>1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ratingValue</td>
<td>int(3)</td>
<td>YES</td>
<td>NULL</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Count</td>
<td>int(11)</td>
<td>YES</td>
<td>NULL</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Average</td>
<td>Float</td>
<td>YES</td>
<td>NULL</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Search_hit</td>
<td>Int(11)</td>
<td>YES</td>
<td>NULL</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 5.1 Field variables for the project system

From architectural point of view, the above datafields are categorized into two types of contextual data: data describing fields and environment describing fields. These two different type of fields are illustrated in figures below.
Figure 5.1 Data describing fields in the database

- Description of data
  - "Desc"
- Geographic Location
  - "locx" "locy"
- Average Rating
  - "Average"
- Location Name
  - "Name"
- Multimedia Description
  - "image"

Figure 5.2 Environment describing fields in the database

- User Location
  - "Address"
- Time and Data
  - "curr_timestamp"
- User Name
  - "User"
- User Age
  - "Age"
- Multimedia Content
  - "image"
- User Profession
  - "profession"
One group of field present in the database of the current project reflect the description of the data itself, while other group of field presents the description about the user which also reflects the operational condition of the user.

The various fields that reflect the context of the data are discussed below.

The “name” field in the database reflects the name that the user gives to a location. For example, in the present project scenario, the user can upload any location to the database which he/she visits. This can be a tourist attraction, a restaurant, a landmark, and so on. Thus, the name of the location belongs to the context.

The “Descr” field is added which shows more information about the location from the user’s perspective. This lets the user define what he/she can relate to with the location. For example, if a restaurant is loaded into the database, the user can add his description about what kind of food it serves, the internal architecture of the restaurant, the kind of service that one should expect if he/she is to visit the place, and so on.

To further support the perspective importance of the context, the “image” field has been included. This field supports the addition of multimedia content to the database. The multimedia content allows the user to give his/her perception about the location as it is easier to visually analyze the point rather than descriptively follow the data. Description about a data not only conveys the local context of the location, as does the description, but also conveys the environmental context of the user, such as time and date, weather conditions, and so on.
The “ratingValue” and the “search_hit” are fields that describe how popular the given information is.

From the above fields, although a lot of information is conveyed about the data, it is very little known about the environment of the user. The context-aware system should consider those variables which give information about the surroundings that also affect the behavior of the user. For example, the behavioral patterns of other users who are related to the present user may affect latter’s choice. Thus, the relationship between the users must be taken into consideration by the context-aware system. Such information is collected in order to include recommendations of similar class or group of people in search results.

To collect information about the user, fields such as “name”, telephone number (“tel_num”), age (“age”), profession (“profession”) are added to the database table. This allows the system to better reach the needs of the user by recommending them locations that are liked by their class of people, or which are liked by people nearer to their geographic location. To relate the searches by a group of people near to their geographic location, the telephone number is utilized, which looks for the choice of people who have the same area code and hence address a general class of people.

Another field that can be added to the description is the way in which the fields can be accessed. That is, if the fields are just meant to be read-only, or can be written by other users about contents uploaded into the database. Thus, for any variable defined as unique inside the data list of field variables, the control access can be defined as ‘α’.
the present system described in chapter 4, the two “ids” that can have the access control as shown in the following table:

<table>
<thead>
<tr>
<th>Variable_id</th>
<th>Access Control (α)</th>
</tr>
</thead>
<tbody>
<tr>
<td>location_id</td>
<td>α ∈ {r,w}</td>
</tr>
<tr>
<td>ratingValue</td>
<td>α ∈ {r}</td>
</tr>
</tbody>
</table>

Table 5.2 Access control of the variables

Since, the ‘location_id’ of the users can be read by different users and written by according to the recommendation by the project application, it has an access control of read and write. On the other hand, the ‘ratingValue’ of a particular location is read by the system and hence have the access control of read only.

The organizational structure of the system database can be explained with the help of certain system parameters. These parameters are variables that the application or system assign with regard to a user. Thus, for several variables which are associated with a particular user, the scope of these variables may or may not extend to data of other users. This leads to division of the variables of the users categories as discussed below.

Local variables associated with a user are variables, which hold values private to the user. These values are not exchanged or viewed by variables of other users. Global variables associated with an agent are variables, which hold values, which are public to other users or programs. These values can be exchanged or viewed by variables of other
users. The global variables are seen by the parameters of other users, and hence they are interdependent on other users.

5.2 Variable Interdependencies

After the system specification is described in detail, and the variables and their users are declared, the reaction of the variables are needed. This reaction specifies how variables can react to changes in other variables. Variable interdependencies of three data-types in a system can be abstractly depicted as in figure below:

![System specification with variables and users](image)

Figure 5.3 System specification with variables and users

The variables that react to changes in other variables, impact the attributes of the user. This means that if a variable is changed, then the context-aware application may change the resultant behavior given that certain conditions are satisfied.

This type of reaction behavior is described in the following figure.
39

In the above figure, the context definition of our system is defined for any arbitrary data belonging to the context group ‘Q’. The context group is a group of users that have similar choice or belong to a common class of people. Thus, for our location-aware system, a particular class of people can be grouped together for a similar choice of locations in a particular area. Hence, if a user has made a particular choice, it is most likely, that another user will make that choice. Hence, all the agents belonging to group Q look for changes in the ‘location_id’ variable. If a particular user has made a choice of ‘location_id’, then the current user checks for the condition for ‘popularity_index’. If the condition for ‘popularity_index’ being greater than a predefined value of ‘a’ is satisfied, then the current location is recommended to the current agent. Such an approach towards recommender system is highly desirable as a group of people dependent upon the choice of one another.

5.3 Design Implementation

As the last step to build a ubiquitous location-aware system, the web interface is developed which acts as a medium to communicate among humans and the computers.
The user can supply the context-based query which can be any of the tags that are described with the stored location. Once the database has been pre-loaded with the entries of locations, the user can retrieve the data by specifying the search criteria. This flow of information is shown in the figure below.

Figure 5.5 Search system interface architecture

The search page asks the user to enter keywords to search from, as shown in figure below. The searching interface also asks specifically the user, which contextual data he/she wants to look into by creating a drop down menu.
In this project, the database is kept with the name of the restaurants near the geographic location of the user. When the name of the restaurant is searched upon, the PHP interface interacts with the MySQL database to search the name of the restaurant. If the data that the user is looking for is found in the database, then the geographic location of the data is compared with the geographic location of the user to calculate the distance using the distance formulae.

The search criterion is based upon the number of ‘hits’ a particular location based database entry has, as well as the minimum distance to the user’s given coordinates. This
system’s search criterion restricts the search within 5 miles of the user’s coordinates. The main reason for restricting the search based upon the nearest distance is because of the spatial locality in the user’s context. If a user is looking for any location, then there is high probability that the user is looking for a location near his/her current location over other locations.

Figure 5.7 Search result for the system

Thus, in this application, the spatial locality closeness of a location being searched is given more preference over any other locations. But, to further refine the result, the most popular criterion is also taken into consideration. Here, if there are multiple
database entries which match the user’s specification for search, then these multiple entries are sorted according to their individual popularity index. This index is continuously updated as more users search for a given location, and is specified in the database by the field ‘search_hit’ as described in previous section. The final location is mapped into the API and the result is shown on the web page as described in the figure below.

Figure 5.8 Mapping API interface

Here, the system 1 calculates the restaurants in the local area, and first sorts out the ones which are nearest to the user. Then, among the sorted out results, it displays the ones which have the highest “popularity_index” among the same class of people. This way, the restaurant location is contextually relevant to the user.
A similar web page is developed for system 2. Here, database is not preloaded as is the case with System 1, but the database is gradually built by users from the web page. Thus, this application is more mobile as the users can update the database with their preferred location from a mobile device.

![Web interface for system 2](image)

**Figure 5.9 Web interface for system 2**

In this system, another functionality of “moving the map” has been added, which allows the user to change the center. Using this feature, the user can move more freely in the map and look for database at other locations.

This system is based upon the average rating for every location in the database. As the users rate a particular location, the average for that rating is changed in a manner similar to the popularity index in the previous system. But, in this system, the rating
based criterion is more sophisticated as it provides more accurate results. The database entry is made directly from the map.

![Image of map with input fields](image)

**Figure 5.10 Input fields for database on web interface**

Thus, the data is directly stored into the database from the map along with the context tags in the ‘Description’ field. The address, latitude and longitude of the data are retrieved from the map through the API. Further, the user rates the location among the indices 1 to 5 and the time and date contextual information are saved along with the database.
Thus, when any future user looks for any location, the data is matched against the previously stored data. The result also shows the location which has the highest average rating which the user can select and also input his rating for the location. This step allows the location based database to be dynamic and change the average rating by itself. Any further multimedia content can also be added by the use of the mapping API functionality. This enables the application to be completely mobile, a feature which was not so efficient in System 1.

The System 2, unlike the previous system, allows the user to choose from a set of recommended solutions and rate the data. This allows that the past solutions of the user’s preferences to be monitored to allow the system to recommend solution to new users in the future extended development of the current system.
Chapter 6

CONCLUSION

This chapter concludes work done while developing the project. It discusses its limitations as well as the strengths. At the end, the future development scope of current system is discussed.

6.1 Conclusion

Through this project system, a small step is taken towards the attempt to imitate the thorough understanding of the real life context based upon, the different attributes supplied by the user, the system is able to justify the search. The system effectively returns the data which closely matches the required tags of the user.

The pre-defined restriction of sorting the data on the basis of least distance from the user provides the use of the system in location-aware applications such as mobile-web applications. The popularity index, which was another attribute to sort the data, was included to ease the user to find the data which was most followed by the community. In the modern context, as the number of available data increases, the popularity index provides an initial way to sort the data out. Further data refining is done by the interdependencies between the users as depicted by the Attributed Relational Graphs (ARGs). Thus, the system was a self-taught for recommending data to the users based upon their needs. It was further found that although the system provides results based upon group activity of the users, but it gives the user the freedom to control his data such
as rating of a location, or obtain a particular search location irrespective of the activity of the group of users.

The organizational framework of the system clarifies the relationship between the ubiquitous computing and mobile systems. Thus, such a system is an essential step in establishing the next generation framework for self-intelligent systems.

6.2 Future Scope

This project, with its small scale implementation of the advanced algorithmic analysis Attributed Relational Graphs (ARGs), is made with tremendous future perspectives. This project can be further developed to improve the relationship attribute between the user’s current location and the other nodes available to the user from its search. As the depth of the multimedia content increases, the number of locations that a user can go to increases, and thus the choice to prioritize the next available search is what makes this study and project a worthy experience.

A major problem with the current system also provides a future scope of development. In System 1, the data mining from the database was accomplished by Boolean matching of the occurrence of the tags. This can be further refined by development of advanced algorithms to establish a relationship between the past and present activities of the users belonging to the same group. Such a system would require advanced probabilistic algorithms which can be integrated into the present system.
This suggestion is made with an interest to inspire the group of people to further develop systems which can make the use of artificial intelligence at the fullest and shift the paradigm from stale desktop applications to mobile and ubiquitous applications.
BIBLIOGRAPHY


