An Ontology based Context-aware Application

Gaurav A. Ahuja
Computer Department
VESIT Chembur-74.

Vishal S. Kotak
Computer Department
VESIT Chembur-74.

Siddharth D. Keswani
Computer Department
VESIT Chembur-74.

ABSTRACT
The paper gives information about Context-Aware recommender application with its data structure implemented in Ontology. The basic idea here is to recommend the user of the application places and services according to his/her preferences and/or context. Ontology is a way of representing data in the form of concepts in a particular domain and the relationships between those concepts.

General Terms
Ontology, Data Mining, Context Awareness, Location based services

Keywords
Context awareness (Location, history), recommender system, Ontology, preferences

1. INTRODUCTION: -
Today, users want to communicate with their smart devices just like the way they want to do with their friends. Just as a friend knows about favorite dish, place to hang out, bank in which his friends hold an account. Won't it be amazing when smart devices make a person feels special just like friends? Like a friend who informs about his friend's favorite store giving a discount, won't it be amazing if o device informs us about the same? The application aims to give the user such an amazing experience.

The first wave of mobile devices provides the basic calling and messaging functionalities. The second wave provided additional simple email and internet functionality. The third wave brought about a revolution by bringing a change in user interface, user experience and the ability to install or remove applications customized according to user needs.

The number of mobile device users have increased exponentially which has led to more data generation on a daily basis. This data when analyzed can help understand the likes and dislikes (preferences) of the user. Using this preferences the application aims at recommending the user about food, places, movies etc. Here, the advantage of the recommender system is it acts like a friend, knows preferences and accordingly informs the user without him/her querying for it.

Whenever the users move to new surroundings, they want to explore the place, and at that time it is difficult for them to find new places matching their preferences. Location is an integral part of the context in which the user is. The application aims at combining the location and preferences of the users to give relevant results.

2. CONTEXT AWARENESS
Context awareness is the knowledge that a system has or gains about the context of the user. Context comprises of user history on the web usage, chat data, applications and user location. Using the user's context there is a creation of user's profile.

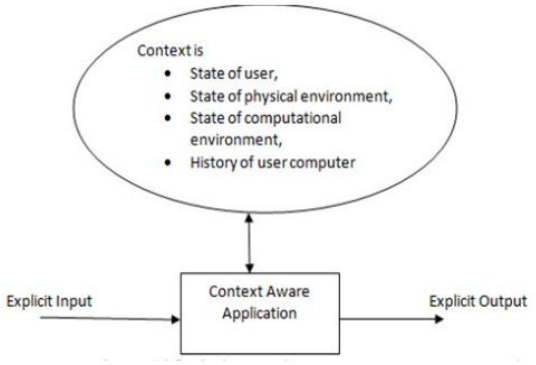

Fig 1: Context Awareness diagram [2]

2.1 History
The tendency of human beings is to search for similar or related objects. Taking advantage of this tendency, application provides recommendations to users according to previous search queries. Below is the snapshot of the device retrieving the history of the user.

As you can see below, the top one gives information about the websites user visits, which helps to record user's preferences. The bottom one gives information about the bank in which user has an account in which will help the application to recommend nearby ATM's when the user moves to a new location.

This information is mapped to the user and during and the recommendation is provided according to users past experiences.

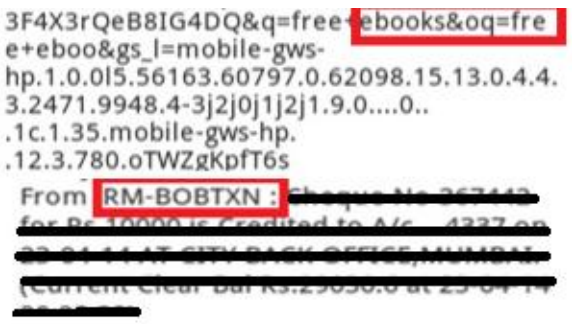

Fig 2: Inbox and web history snapshot

2.2 Location

Position of the user is the latitude and the longitude {x, y} co-ordinate relative to earth. Position can be measured using the GPS or the cellular network. But the context of user is the surrounding area i.e. radius of 5-10 Km around the position. This radius is called as reach.

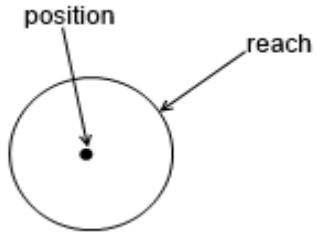

Fig 3: Location = position + reach.[1]

3. ONTOLOGY

Data is the most important aspect in today's world. All the industries in the world are hungry for user data. They want to analyze the spending patterns of the user and also their preferences. This information helps them to sell their product and target a certain set of audience for that product. Products are designed according to the user needs that yield more profits.

Managing huge volumes of user data is an uphill task. The behavior of data is also dynamic. This data needs to be classified and properly structured. Present data management techniques fail to store data in an effective way that cater to constantly changing data and the relations between that data. Here is where ontology helps in data storage, classifications and building relation between that data.

Ontologies consist of concepts and relations which are used to describe data and represent it in area of concern.[8] Ontologies may help to create a structure, store data adhering to this structure, characterizing relationships between this data and adding constraints on this data [9][10]. Ontologies can be complex (consisting of many concepts) or simple (consisting of one or two concepts.).

A general example may help. A bookseller may want to integrate data coming from different publishers. The data can be imported into a common Resource Description Format (RDF) model, example, by using converters to the publishers' databases. However, one database may use the term "author", whereas the other may use the term "creator". To make the integration complete, and extra definition should be added to the RDF data, describing the fact that the relationship described as "author" is the same as "creator". This extra piece of information is, in fact, an Ontology (or a vocabulary), albeit an extremely simple one.

4. IMPLEMENTATION

The main aim of application is that, we didn't want the user to fill forms and give preferences, which according to research is antiquated. The only thing user needs to do is install the application. As soon as he registers with the application, the ontology is created which helps in further recommendations.

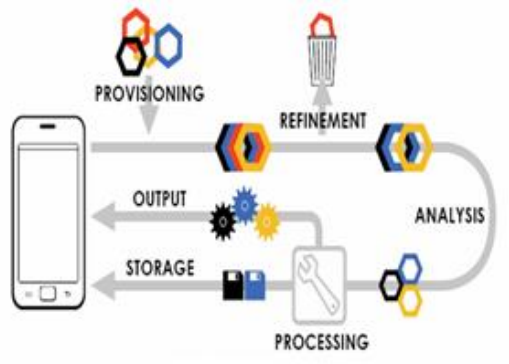

Fig 4: Workflow of the application [3]

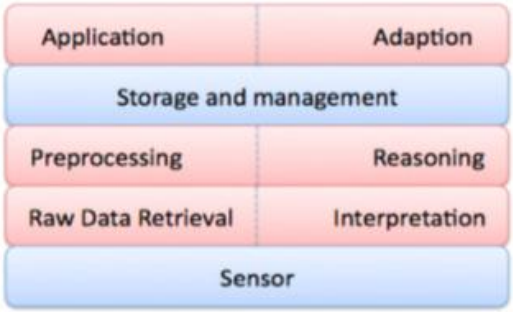

Fig 5: Layered architecture of system [4].

The application works in two phases: -

1. The Ontology phase for data retrieval.
2. The Location phase for context awareness.

4.1 The Ontology Phase

The Ontology which is called the user profile ontology helps mapping of information. According to the requirements of the application, it consists of user preferred restaurants, user's bank name, types of movies etc. The main purpose of using Ontology over database is faster retrieval of data and better classification. For example if a user prefers more than one restaurants retrieved from his past selections, history etc. The database maintains two tables one for user and the next one for restaurants. The tables are then linked through referential integrity (foreign keys) to provide relevant users to the user. This process is time consuming for huge volumes of data. Ontology works on the principle of subject, predicate and object. In our context, the user is the subject, the predicate can be hasBankName or hasRestaurantPreferred and the object are the values associated with the properties.

When the user asks for restaurants, a query is passed to the server considering the information stored in the ontology.

```
<rdf:RDF
xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
xmlns:user="http://example.com/user"
>
<rdf:Description rdf:about="http://example.com/user/userName">
<user:fullname>UserFullName</user:fullname>
<user:restaurantName>ABC</user:restaurantName>
<user:restaurantName>DEF</user:restaurantName>
<user:restaurantName>GHI</user:restaurantName>
<user:bankName>POQ</user:bankName>
<user:bankName>ASD</user:bankName>
</rdf:Description>
</rdf:RDF>
```

Fig 6: RDF graph

The above code shows an ontology in RDF format.[12] Here when the query is passed to the server, the names ABC, DEF is given a consideration and the results are shown which may be the most relevant to the user.

We have used androjena API developed by HP to integrate ontology with Android.

```
Model model = ModelFacory.createDefaultModel();
String ns = "http://example.com/user";
model.setPrefix("user",ns);
```

Fig 7: Model also called as graph

The above code helps to create an RDF graph which is the source of data.

```
Resource user = model.createResource(ns + UserName);
```

Fig 8: Creating a resource

Resource in simple terms relates to subject i.e. the user for our application.

```
Property p1 = model.createProperty(ns,"restaurantName");
Property p2 = model.createProperty(ns,"bankName");
Property p3 = model.createProperty(ns,"movieName");
```

Fig 9: Adding properties to resource.

These are the attributes associated with the user. Now comes assigning values to these properties. This is done by his history as well as past selections in application. It also depends on the ratings which the user assigns to a particular place. The rating is completely subjective and will be taken into account as per the individual. We have used a simple text search algorithm to find the details from history as everything is stored in the form of text strings.

As soon as we receive some information we will add it to the ontology depending upon the rating. This can be done using following code snippet.

```
user.addProperty(p1,"ABC");
user.addProperty(p2,"POQ");
```

Fig 10: Adding values

The mandatory thing which needs to be done is we need to import some JAR files which are given in the androjena documentation. The main advantage is that we can add multiple values to the same property thereby increasing access speeds and reducing data redundancy.

4.2. Location Phase

We have implemented application for Android OS. Android is an open source mobile operating system (OS) based on the Linux kernel and currently developed by Google. The application features login and register system. To benefit from system, users need to first register with the application. Facebook and Google + users can directly login using their Facebook login and Google+ accounts. Other users can use their email ids to register. The once for all login can be used by users who don't wish to login each time they want to use the application.

Users can search for nearby places using different categories of restaurants, ATM, coffee, hospitals, etc. Each search request returns 20 places using the Google Places API. The places are displayed in list View with option of loading more places. The application also provides a Map View using the recent GoogleMapsAPI V2. Users can toggle between List View and Map View using a simple Swipe. There is also a Filter page provided to the user which can be used to refine the places according to the different options provided. On clicking each place, user can get detailed information about the place along with route to that place. Application will automatically provide recommendations, in the form of notification, whenever the user location has changed. User can turn off the notifications anytime he/she wishes using the settings options provided.

Now comes the HOW part of the application. The fact that it's a location based app, calls for many different problems. To tackle all the different problems taking utmost care that the user experience is never compromised, was a challenging task. The most common problem which location based apps face is the delay experienced while they are waiting for the GPS to get a fix. In our application is has been tried to minimize this delay as far as possible. In android there are two categories of location provider's fine providers and coarse providers. GPS comes under fine location provider while network provider comes under coarse location provider. Nonetheless, accuracy comes with time (and also with significant impact on battery life).Though the GPS provider is accurate to as low as 3 to 4m, it is very time consuming and drains the battery to a considerable extent. Network provider on the other hand, is fast but less accurate.

In the application, a less commonly used strategy which takes into account the location results obtained by other apps in our mobile phone is used. This location provider is the Passive location provider which takes the least amount of battery along with the lowest time. For e.g. GoogleNow periodically tracks our location, so this Passive provider will take location results from GoogleNow. The advantage of this approach is that the user doesn't have to wait to get his/her location tracked. The time in which the nearby places will be obtained will only depend on the network connection.

Now, the location obtained from passive provider is sometimes accurate and is sometimes not. So completely relying on that is possible. To tackle this problem, background service as soon as user opens the application is executed. This service will request a single update from the location manager. This location manager will take the location from either the GPS provider or network provider depending on whether the GPS is on or not (We are leaving GPS usage to be solely under the discretion of the user).As soon as the location manager obtains the location fix it will call the onLocationChanged() method which will return the location to the android activity. On receiving the new location, the list of nearby places will get updated and all this will happen without interrupting the user.

The last problem with location based apps is when to update the list of places. By rigorously experimenting with the Google Places API at different places, a conclusion has been reached that the places do not change in a radius of at least 200m around the user. So we refreshing places only if the user location has changed by 300m.This distance is safe enough to cover all the nearby places without missing out on even a single place. As soon as the user exits the application, this service gets stopped.

5. DIAGRAMS

5.1. Use Case Diagram

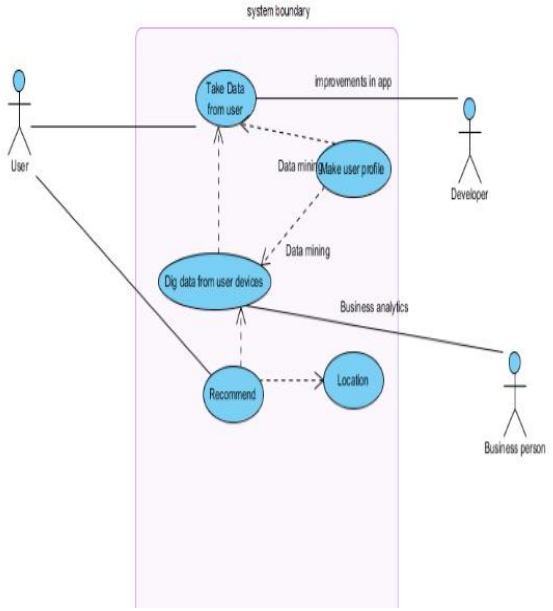

Fig 11: Use case diagram

5.2. Class diagram

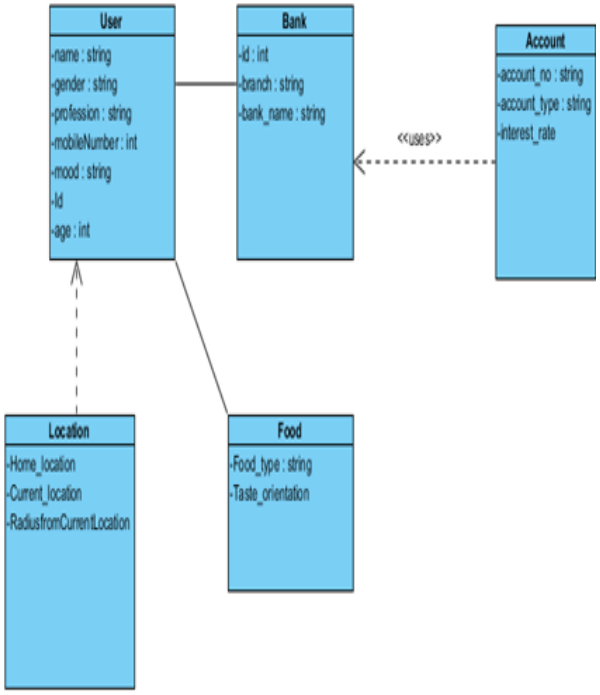

Fig 12: Class diagram

6. SCREENSHOTS

Fig 14: List of restaurants

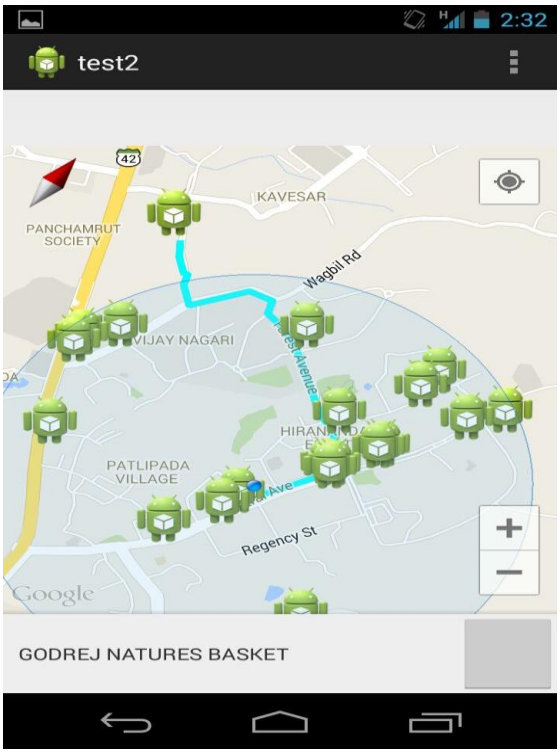

Fig 16: Map View

The fig 14 show us a list of restaurants, considering users history through ontology as well as current location, both are part of context aware applications. Fig 15 displays it in the Map View so that the user can decide which destination he prefers, depending upon the route, transportation as well as the time taken to reach.

The application also focuses on emergency services such as hospitals, where a user can find the nearby hospitals, contact them and ask them for an ambulance, fastest route to reach, the services provided by the hospital and the specialization.

It also focuses on providing exciting offers to users frequenting some restaurants, as soon as they are in a range of about 500m from that restaurant.

7. LIMITATIONS

The application has some limitation which will be surely looked after in the nearby future. One of which is this application needs a smart phone device. According to us, this limitation is already being solved by cheaper smart phones. The location grabbing process sometimes drains battery in some devices as this application runs in the background.

The main concern of our application can be the privacy issue as users need to trust us with sensitive information. The end user agreement of Android and iOS provides the developer with the rights to access user information to be used in a proper way without leaking it to unauthorized sources and keeping care of the user privacy.

8. FUTURE SCOPE

1. Our main aim is to bring sentiment analysis into the application where the recommendations will be given according to the user's current mood.

2. The Application can be converted in future to a social network recommendation application, wherein the preferences from the peers and relatives, can also be provided and the user can take advantage of those suggestions.

9. REFERENCES

[1] Gaurav Ahuja, Vishal Kotak, Jeetu Rijhsighnai. Mode Manager (Oct 2013) IJERT. VESIT

[2] Anind K. Dey, Daniel Salber, Gregory D. Abowd, Providing Architectural Support for Building Context-Aware Applications.

[3] Alf Inge Wang,CAMF – CONTEXT-AWARE MACHINE LEARNING FRAMEWORK FOR ANDROID

[4] M. Baldauf, S. Dustdar, and F. Rosenberg, "A survey on context-aware systems," Int. J. Ad Hoc Ubiquitous Compute vol. 2, pp. 263-277, and 2007. M.Miraui, C.Tadj."Architectural survey of context aware systems in pervasive computing environment," Ubiquitous Computing and Communication Journal. Vol3 2008.

[5] Anind K. Dey, Daniel Salber, Gregory D. Abowd, Providing Architectural Support for Building Context-Aware Applications.

[6] T.R. Gruber, A translation approach to portable ontology specification, Knowledge Acquisition (1993).

[7] M. Uschold, R. Jasper, A Framework for Understanding and Classifying Ontology Applications, in: Proc.IJCAI99 Workshop on Ontologies and Problem-Solving Methods, Stockholm, 1999.

[8] Winograd, T. (2001) 'Architectures for context', Human-Computer Interaction (HCI) Journal, Vol. 16, No. 2, pp.401–419.

[9] Gediminas Adomavicius**,"**Context Aware Recommender System".

[10] B. Schilit, N. Adams, and R. Want, "Context-aware computing applications," in Mobile Computing Systems and Applications, 1994. WMCSA 1994. First Workshop on. IEEE, 2008, pp. 85–90.

[11] BOTTARI: an Augmented Reality Mobile Application to deliver personalized and Location-based Recommendations by Continuous Analysis of Social Media Streams.

[12] E. Dumbill. Finding friends with XML and RDF. In IBM developerWorks, XML Watch. xmlhack.com, June 2002.