Location Aware Question Answering based Product Searching in Mobile Handheld Devices

SK Alamgir Hossain, A. S. M. Mahfujur Rahman, Thomas T. Tran*, Abdulmotaleb El Saddik
Multimedia Communications Research Laboratory
SITE, University of Ottawa, Ottawa, Canada
Email: (skahossain, kafi, abed)@mcrlab.uottawa.ca, *ttran@.uottawa.ca

***Abstract*—In this research we present a question answering based searching technique for location based shopping. A user may ask a question to the system as they naturally ask to a human while the system retrieves the search results by analyzing the given question and the current GPS location. Based on the retrieved results, the system carry out conversation with the user to explicitly understand his/her needs and accordingly filters search results for display. The conversation between the system and the user is based on word co-occurrence keyword extraction and Artificial Intelligence Markup Language (AIML) technique. As per initial experimentation, we found out that the proposed approach of conversation based shopping is appealing and useful to the user.**

***Index Terms*—Location aware computing, computer agent, virtual shopping, conversation system**

I. Introduction

Presently the usage of mobile phones is increasingly becoming popular in public places. The emergence of mobile phone location based services [1][2][3] like product searching, viewing, and purchasing are the key developing areas in Mobile Information Technology [4][5][6] that introduces electronic marketplaces instead of physical one [7]. Now people prefer online product checkout rather than to go to a physical store. A mobile user can use the handheld device to carry out searching operations in the marketplace and later view the results in the mobile screen. Online product comparison, purchasing, bill payment etc could be done using the mobile devices easily. However, now-a-days the number of electronic commerce based sites is so large in the Internet that with a simple query it is very difficult to extract the needed information. In the web product brokering sites [8][9][10] are available for this purpose to assist the users in the searching process. The product brokering site searches different e-commerce stores for products in order to obtain the best results by using the user's query. Indeed, if web and Internet services run in this manner then in future the physical shopping will be replaced by online shopping.

According to current search technique [11][12][13] users are required to provide some keywords in the search field to search for any information. The search engines usually do not accept natural language like query even though users often prefers to interact with a site by using natural language like query processing system. Zadrozny et al. [14] agreed that the best way to facilitate Human Computer Interaction (HCI) is by allowing users "to express their interest, wishes, or queries directly and naturally, by speaking, typing, and pointing". A myriad e-commerce based sites are available from the mobile devices to search and list products online . Many of these e-commerce sites provide personalization agent to retrieve products of interest from the site and assist the user in product selection. The agents employs domain specific natural language based query processing and calculates the keywords to intelligently process the query.

When search results are displayed in the browser page, the user further needs to manipulate the filtering options in order to refine the obtained results. This repeated interaction for filtering the displayed results can be minimized if the system employs natural language processing based user query processing system. By analyzing the obtained search results, the intelligent query processing system can use question answering technique in order to determine the filtering options for the users. For example, if a person requires information about a product and may inquire about it from a human or from a search engine in a computer. The responses from the human can be better than that of the computer program. If we reflect about the reason we realize that in case the human didn't understand the question or if the answer depends on many other parameters then the human responder can further ask questions and after getting clarifications of those parameters a refined response could be prepared for the questioner. I.e., human can make some questions or request clarifications to the questioner and based on this human-human conversation the answer may prove to be more accurate. If we apply this experience to human-computer interaction then the search results will be more appropriate than a single query. Moreover often the user has limited idea about the actual search query that needs to be placed to the search engine to get the results that s/he wants. So if we design our query processing system to mimic a human-human conversation, accessing of the search results can be improved significantly.

So, a personalization based intelligent query processing is required to assist the mobile user in locating relevant search results easily. The contributions of this research is that, we propose a system in which the mobile user can make conversation with an intelligent agent, installed in his/her mobile device and search a location specific product information. The proposed method can easily be incorporated in a personal online auction, electronic market place, product comparison sites, and wireless advertising etc. The remainder of the paper is structured as

follows. Section II describes some well known related works in this area. Section III illustrates the core architecture of the proposed system. We developed a prototype of the proposed model, which is discussed in Section IV. Next Section V presents some evaluation of the implemented prototype and describe our findings and finally conclude the paper in Section VI.

II. Related Work

Location based services like location based shopping is not new in electronic commerce. The first notable work is Shopper's Eye [11], which is a PDA based location enabled agent prototype. This prototype is used for supporting shopper while they are shopping in a mall or in a store. This agent prototype just provides information related to the users previous interest. When shoppers are in a shopping mall, this agent notifies him or her of the availability of products in the current store as well as some cheaper alternatives in the surrounding stores.

Most of the time people have some idea before leaving home for shopping, but the purchase does not correspond to the purchase that brings home from the shopping mall. Researcher [15] try to find the reason of this behaviour and found that 88% of the purchases were because shoppers found products that were offered at a good price or on sale. This type of purchase that shoppers haven't planned to buy is called impulse purchase. This impulse behaviour can be applicable in the location-based advertising where buyer's mobile device connected to an online information system and act as a shopping guide. Many researchers then support this impulse behaviour. One of them is [13], which is a multiagent based prototype which can work in push and pull mode, in push mode merchants could push offer to the clients, on the other hand in pull mode the agent can pull information from an online information system. The main problem of this prototype is that it works only based on some users preferences. Another promising work in impulse is [12] where consumer may direct a PDA-based agent to begin a negotiation with the present store or simply add the product to a "want list" to enable ongoing negotiations and a later purchase. At any time, the consumer may add products such as a "bag" or a "watch" to a "want list". And when he physically visits the location the agent automatically display the information which is matched with the preferences.

Many researchers also attempt to improve the shoppers shopping experience while they are inside a shopping mall. A notable work is MyGROCER [16], which is an m-commerce and RFID based shopping system. In their proposed system an RFID receiver is placed into the shopping cart to uniquely identify each products. The smart shopping cart automatically sends the shopping list to the check-out system and the customers can avoid waiting for scanning the products in the check-out points. Later on a payment receipt is issued to the customer.

Further examples of work to support shoppers anytime anywhere, location-based reverse auctions [17], mobile online auctions [18], GPS based location-based services [19], e-parking [20], wireless advertising [21].

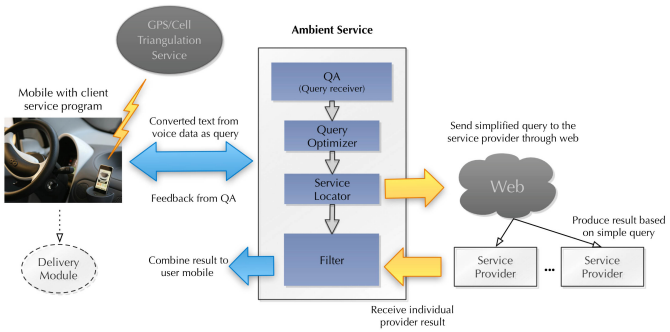

Fig. 1. Core Architecture of the Proposed Model.

III. Proposed Ambient Service Architecture

Fig.1 describes major components of the proposed system. In this system there are mainly two modules, one is the client agent module (here we call this agent as Talkme agent) and another one is the ambient service module. Talkme agent is installed in the client device and the ambient service module is installed in a remote server, which works as a web service. In the beginning in Section III-A we discuss various operations of the Talkme agent module. In Section III-B we describe the Ambient Service module in detail.

A. Talkme Agent Module

To get location based search user needs to install Talkme agent in the mobile device. Primarily Talkme agent performs the following task:

1) Store user profile information: For a new user Talkme suggest the user to provide some basic information like *name*, *sex*, *age*, *favourite color* etc. Talkme uses this personalized data to filter the search data as well as to carry out conversation with the user. For example a user may ask the system *I want to buy a car*, if the user already stored that his favorite color is *black* then the system may further ask *Is your preferred color of the car black?*
2) Conversion of voice to or from text: Talkme also acts as a converter. It performs both speech to text and text to speech conversion. It can take both voice and text data from the user at the same time. We synchronize the input based on time that is, if a user talks and types something at the same time then the system will get those information from voice or text data which comes first, that is, it works based on time. All the voice data is converted to text by speech recognition system and the converted text message is then sent to the ambient service module. After search operation, Talkme receives search results as text, which is again converted to voice. The text will display in the mobile screen and synthesis

```
<category>
  <pattern>I NEED TO BUY A <set name="*"/></pattern>
  <template>Good, </template>
</category>
<category>
  <pattern><check sex="*"/></pattern>
  <template>That is its a <get sex=""/> <get name=""/>?</template>
</category>
<category>
  <pattern><check brand="*"/></pattern>
  <template>I think <get brand=""/> brand is good/></template>
</category>
<category>
  <pattern><check price="*"/></pattern>
  <template>what would be the price limit?</template>
</category>
```

Fig. 2. Sample Artificial Intelligence Markup Language Patterns.

voice played back to the user in the mobile handheld device.

3) Receive mobile position data from GPS: Talkme uses current GPS data in order to determine the current location of the user's device. The signal strength of GPS is not powerful enough to penetrate a building [22]. Hence it does not work in indoor locations. To overcome this problem the proposed system takes advantage of the previous successful GPS data. For this reason, Talkme periodically samples GPS data and stores it in the local memory. So when it is unable to get GPS data, the Talkme obtains the immediate previous GPS data. This is not 100 percent accurate but the last data obtained provides some approximation that points a place and produces adequate clue to the Talkme agent.
4) Communicate with the ambient service: When the user wants to search anything with the help of the Talkme agent, the agent needs to communicate an initialization command like *New*. It then sends a Hello message with the position (GPS) data to the query receiver (QA) module so that the QA module can initialize a new task. Further, the QA module acknowledges to the Talkme agent indicating that it is ready. This step is called connection step. If there is any communication problem like *service not available* or *server is busy* etc that occurs in the initialization process then Talkme notifies the user as well. If the connection is successfully established then Talkme begins the conversation with the user. Every voice data that is received by the Talkme from the user is first converted to text format and then sent to the QA module.

B. Ambient Service Module

This is the main module to process the actual query. This module takes location specific search data from the search providers. It starts its job when a successful connection established with Talkme. Its then continually gets conversational text from the user and feedback the user if more filters are possible. This module receives the text query from the user, optimize the query to get better result, select the appropriate service providers based on the location information which is collected from the Talkme agent and finally search the optimized query to the web server and filter the result if necessary and then send it to the user device. The module consists of QA, Query Optimizer, Service locator and Filter sub modules.

1) Question Answering (QA): This is the first sub module of the ambient service module which has the entire question answering logic to make a human like conversation. For every question QA sub module reply a human like answer or counter question to the Talkme which is happen normally in two people conversation. The conversation is performed by standard AIML [23] (Artificial Intelligence Markup Language) based pattern matching system. AIML is a XML (Extensible Markup Language) compliant language which has mainly consist of *pattern* and *template* pairs. The most important units of AIML are as follows:

- <aiml>: The tag that begins and ends an AIML document.
- <category>: This tag is the fundamental unit of knowledge that contains two or more elements. The least two elements that every category contains are *pattern* and *template*.
- <pattern>: A sequence of characters that want to match one or more user inputs. A pattern (see Fig. 2) may match only one input or several inputs. If a pattern is *"I NEED TO BUY A WATCH"* then it only match *"i need to buy a watch"* by ignoring the case. But pattern may also contain wild card, which matches one or more word. A pattern like *"I NEED TO BUY* $*$*"* will match an infinite number of inputs like *"i need to by a watch"*, *"i need to by a car"* etc. We can also store some information for future use. Consider the first pattern in Fig. 2 which store all the words after *"i need to buy"*, for example if input is *"i need to by a watch"* then the words *"watch"* will be store a variable name *name*.
- <template>: It can specify the response to a particular matched pattern. In template we can use some feature like we can call previously saved value of a variable, we can check whether some portion of a pattern is fall in a class or not. Consider in Fig. 2 there is a pattern which contain *<check sex="">*, this means that when any input is try to match this pattern it first try to check whether it is a sex word or not that is if the input is either *"Men"* or *"Women"* then the input will matched with this pattern successfully. When the pattern matched successfully it's then compile the template value. In the template portion may contain more sub tag like *"That is its a <get sex="">* *<get name="">"* which means that it assign previously stored value which name is *sex* and *name* respectively. That is if the input is *"i want to buy a watch"* then the word watch will be store in *name* variable. And later on when found the keyword *"men"* then it will store *sex* variable. So after that the template value will be *"That is its a men watch?"* and the system will send this value to the user. In this way the system can generate the next question with the help of previously stored pattern variable and the keywords

```
Talkme: Hi John. How can I help you?
John : I need to buy a watch

Talkme: Good, is it for you?
John : Yes

Talkme: What would be the price limit?
John : Not more than 150 dollar

Talkme: Nice, Have you any more preferences?
John : No

Talkme: Ok, I am just preparing your list please wait.
```

Fig. 3. Sample Conversation between Talkme and the User.

found in the search result.

So when QA receive any search query from Talkme it immediately respond to the user if any matched pattern found in AIML, after that it send the query to the query optimizer. A sample conversation with Talkme is shown in Fig. 3 and the corresponding AIML is shown in Fig. 2.

2) Query Optimizer: This sub module receives a user query from the question answering sub module, analyzes the text and finally sends the optimized query to the service locator sub module to get the search result. Algorithm 1 shows the query optimization algorithm. The algorithm takes the user conversational queries as input and returns a list of optimized query text. In Algorithm 1 in the line number 4 the query optimizer get the next conversational text from the QA and then in line 5 determine the keywords in the text. The main vocabularies that are used in the topic are given as keywords. Normally in a noun phrase contain most of the keyword of a sentence. Here we can use existing natural language tools [24] [25] for extracting the keywords from a text data. Here in Algorithm 1 there are two types of keywords are used. First one is matched keywords which are those keywords by which the search data already filtered. In other words matched keywords are those keywords which found in user conversation. Another one is unmatched keyword which are not found in user conversation but found in the search result. That is we have a option to filter the search result with this unmatched keyword. In Algorithm 1 The function getSearchResult() get the search result with the help of the service locator and filter sub modules. In the search result determine the keywords based on word co-occurrence [25] algorithm. If there are *m* number of keywords in the user query and *n* number of keywords on the search result where n $\geq$ m if the search result is not empty. So some common keywords available in user query and the search result because the search result come based on the user query. If the number of common keywords found between the user query and the search result is *m0*. Then we can calculate by Equation 1 the number of unmatched keywords available in the search result. Determination of unmatched keyword is shown in Algorithm 1 from line number 9 to 13. Every iteration of the while loop processed one user query at a time. If the number of conversational iteration is L then the total number of matched keywords will be P (see Equation 2). We call this list is optimized keywords list. In every iteration of the while loop we get some unmatched keywords from Algorithm 1.

Algorithm 1: Query Optimization Algorithm

Input: Set of user conversational queries and current location data
Output: Optimized text query list

```
1  begin
2      P ← null
3      while query remaining to process do
4          query ← getQuery()
5          m ← determineKeywords(query)
6          add the new matched keywords m to optimized
           keyword list P searchResult ←
           getSearchResult(m, location)
7          n ← determineKeywords(searchResult)
8          U ← null
9          foreach Keyword k in n do
10             if k is not exist in n then
11                 add in the unmatched keyword list U
12             end
13         end
14         selectAIMLPattern(U)
15         if user satisfied with the conversation then
16             return P
17         end
18     end
19 end
```

$$U = n - m_0; \tag{1}$$

$$P = \sum_{i=1}^{L} m_i \tag{2}$$

3) Service Locator & Filter: Finally the filter module filters the result if anything require. For example adult content or number of result etc. User can set this preference values to the system. The service locator module at first determines the list of service provider based on the location and the search criteria. Send the optimized query to the service providers. Finally all the result is received by the filter module and combines the result and sends the combine result to the user agent. If the user wants to purchase the product then he needs to choose the purchase method and the delivery option which is an optional choice. How the different modules of the proposed system interact to each other is shown in interaction diagram (see Fig. 4). Here Talkme first initiate a task by sending a *new* command to the QA and then QA will reply the Talkme with a clientID, rest of the time Talkme will have to communicate with the QA by using this ClientID. Talkme then send the first user question to QA. QA will reply if match with any AIML pattern also send the question to the query analyzer. Query analyzer extracts the keyword from this and fetches the

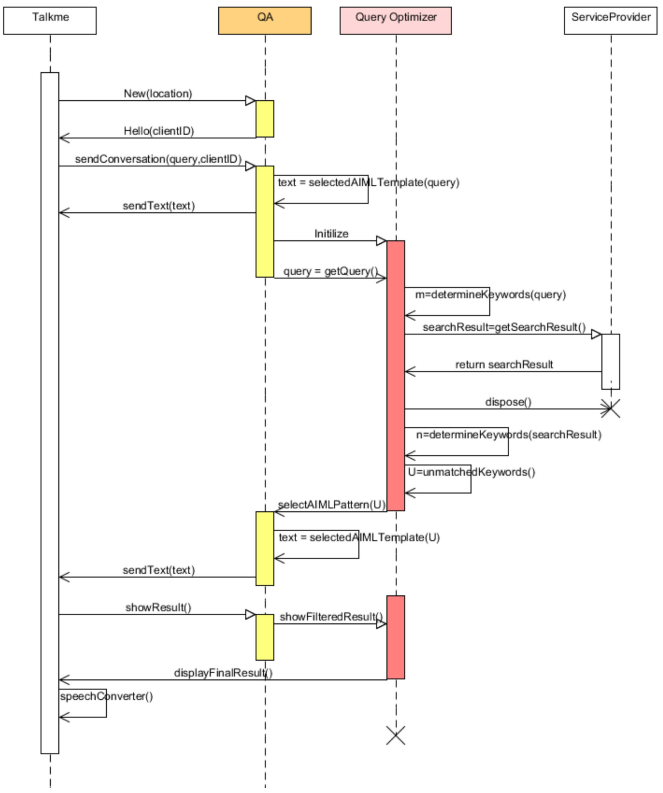

Fig. 4. Proposed System Interaction Diagram

search result from the search provider. Then it determines the unmatched keyword and sends random unmatched keywords from them to the QA. QA then select and send to Talkme an appropriate template based on the pattern match. When the user again respond it will again send to the query analyzer in the same way as discussed above but only difference is now query analyzer need not to fetch the search result again from the search providers instead of this it use the previous search result to filter more. In this way the system filter the data until user express interest to see the data that processed or there are no unmatched keywords available.

IV. Trial Prototype

In order to check the validity of our approach we partially implemented a prototype of the proposed system. We implemented both our proposed Talkme agent for client mobile device and ambient service module for the web service. This prototype only for some simple conversation and the prototype use locally running web service instead of actual web service running in the Internet. Here local web server use a simple XML file which act as a data store for the service. All the product details with the location information are already in this file. In the actual implementation need not to store the product information in a XML file. In that case we can use database server like SQL server or MySql or any other database. Here we used XML to minimize the implementation time. A sample conversation and the final output of the prototype is shown in Fig. 5 which use the AIML file described earlier in section

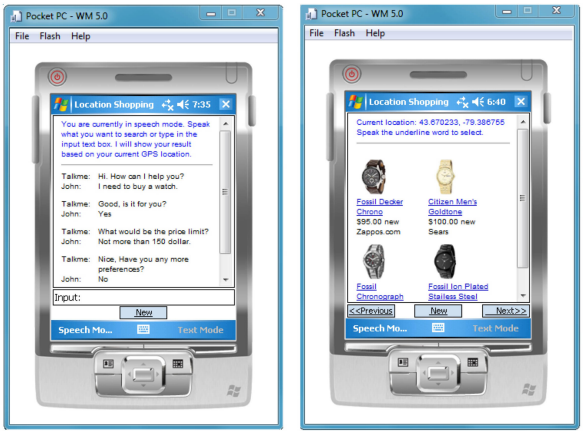

Fig. 5. Talkme Beta Prototype.

III-B. The prototype has two working modes: text mode and speech recognition mode. In the speech recognition mode we can speak anything which converted using Microsoft SAPI [26] speech recognition SDK and in text mode we want to type the text conversation. So some older phone which has no speech recognition facility can be able to work with text mode.

A. Technical Issues

We used the following existing software tools or API for the prototype.

- Microsoft Visual Studio 2008 with C#
- Microsoft Pocket PC Emulator Version 5.0
- Microsoft Speech API (SAPI 5.0 SDK) [26]: The SDK contains both a speech recognition engine and a text to speech engine. The speech recognition engine is capable of large vocabulary continuous speech recognition and context free grammar (CFG) based recognition.
- OpenNLP [24]

V. Evaluation

We have performed quantitative and qualitative measurement studies to evaluate the user's quality of experience with the prototype and to justify the suitability of the proposed approach. The quantitative analysis of the prototype is performed by evaluating the search performance. On the other hand, the qualitative analysis is performed by studying different usability aspects of the proposed system. These are described in detail in the following:

A. Quantitative Measurements

Fig. 6 shows the search performance with respect to different test cases. The test cases are shown in Table I. Each test case indicating the first question of a search and the rest of the conversation will happen based on this initial question. Two user test the system using each test case and their result also

TABLE I
TEST CASES FOR TESTING QUESTION ANSWERING MODULE

Test Case #	Question
1	I need to buy a watch.
2	I want to buy a citizen watch.
3	Give me some netbook price cheaper than 400 dollar.
4	I want to buy a citizen men watch.
5	I need to buy a personal computer.

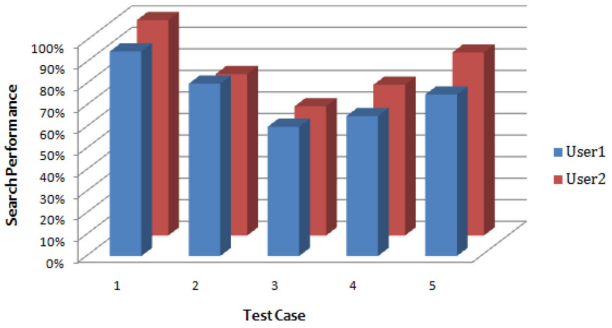

Fig. 6. Test Case vs. Search Performance.

shows in two different colors in Fig. 6. The search performance measured by using Equation 3.

If the total number of item that relevant to the conversation is α and total number of search result displayed in the mobile screen β then the search performance η

$$\eta = \frac{\alpha * 100}{\beta} \tag{3}$$

B. Qualitative Measurements

We have performed usability tests to qualitatively measure the proposed system. The usability test consists of ten volunteers of different age groups and academic backgrounds. The users were requested to use the developed prototype. Based on their searching experience they were asked to answer several questions. The answers to the questions are in the range of 1-5 (the higher the rating, the greater the satisfaction) on a Likert scale [27]. Fig. 7 shows the users' responses in percentages. The responses show the suitability of the proposed system.

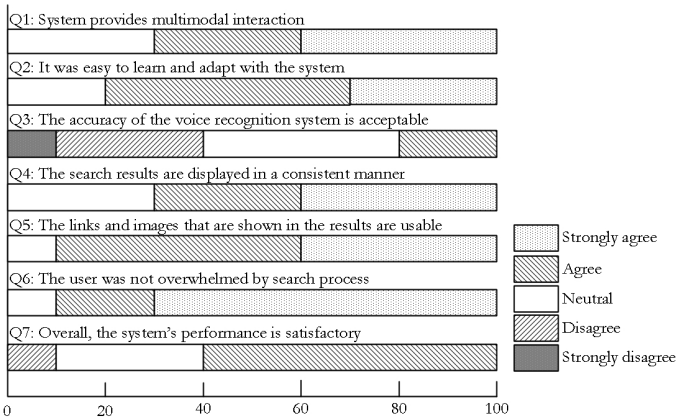

Fig. 7. User Response Percentage on the TalkmeBeta Prototype.

C. Discussion

After the quantitative and qualitative measurement result, we have identified numerous research challenges regarding the possibility of commercial deployment for such systems. First main problem is currently the speech recognition technology is not so much intelligent that it can recognize all the speech text correctly. Also the performance for keyword extraction from sentence is not good enough. The performance is only 57%. According to qualitative measurements result (Fig. 7) most of the user (80%) think that the system performance is mainly depend on the accuracy of the speech recognition. Although most of the person (60%) agree that the system performance is satisfactory. From the quantitative measurement we found that for simple test case the system performance is good. For example for test case # 1,2,4,5 the average search performance is about 80% or greater. Only for some test case like test case # 3 the average search performance is about 50%. This is a low percentage because the prototype fails to identify the exact keywords from the sentence. Although our system has this problem but it has the following important benefits:

- Natural language like searching facility.
- In order to formulate a search query the user produces some keywords in the form of conversation with the system. Later on the system finalizes the query by using the question answering system.
- User can personalize his or her search result.

VI. CONCLUSION

In this paper, we presented a new approach for visualizing location based search results in mobiles or PDA devices by using natural language based question answering system. We focused mainly on the location based shopping, however the scheme can also be incorporated in many generalized e-commerce based searching technique specially in e-business. In our prototype one of the limitations of the system is that the question answering module uses the previous answers to further concise the product search lists. In the scheme the QA module is dependent on the previous answers and hence if the user needs to change his/her search domain then it is needed that the user should start over from the beginning. Basically, as the search results come from the service providers based on the first query and then the QA module subsequently filters this result to match user needs. Therefore, in order to start a new search the user needs to put his or her main search item in the first question. In our future work we want to address the issue into more detail. However, we believe that our proposed location based search techniques will remain as a motivation for further research in this area.

REFERENCES

[1] B. Rao and L. Minakakis, "Evolution of mobile location-based services," in *Communications of the ACM*, vol. 46, no. 12, December 2003.

[2] M.-A. Aufaure, S. Yu, S. Spaccapietra, and N. Cullot, "User profiles in location-based services: Make humans more nomadic and personalized." in *In Proc. IASTED International Conference on Databases and Applications*, 2004.

[3] J. Schiller and A. Voisard, "Location-based services," in *Mor-gan Kaufmann*, San Francisco, California, 2004, p. 250.

[4] A. Tsalgatidou, J. Veijalainen, and E. Pitoura, "Challenges in mobile electronic commerce," in *Proceedings of IeC 2000. 3rd Int. Conf. on Innovation through E-Commerce*, Manchester UK, 14, Nov. 2000.

[5] G. M. Chinnery, "Emerging technologies going to the mall: Mobile assisted language learning," in *Language Learning and Technology*, vol. 10 (1), January, 2006, pp. 9–16.

[6] T. Magedanz, K. Rothermel, and S. Krause, "Intelligent agents: An emerging technology for next generation telecommunications," in *INFOCOM'96*, San Francisco, CA, USA, Mar, 1996.

[7] Y. Bakos, "The emerging role of electronic marketplaces on the internet," in *Commun. ACM*, vol. 41, 8, Aug. 1998, pp. 35–42.

[8] GPSHOPPER-LLC, "Mobile local product search for retail shopping, http://www.slifter.com."

[9] eBay, "ebay: Online marketplace that enables trade on a local, national and international basis, http://www.ebay.com."

[10] Amazon, "Amazon: American-based multinational electronic commerce company, http://www.amazon.com."

[11] A. Fano, "Shopper's eye: Using location-based filtering for a shopping agent in the physical world," in *In Proceedings of the International Conference on Autonomous Agents*. ACM Press, 1998, pp. 416–421.

[12] J. Youll, J. Morris, R. Krikorian, and P. Maes, "Impulse: Location-based agent assistance," in *In Proceedings of the Fourth International Conference on Autonomous Agents*, Barcelona, 2000.

[13] R. Brena, L. Fernandez, J. Dominguez, and J. Aguirre, "Location-based support for commerce using multiagent negotiation," in *Research on Computer Science*, vol. 17, Mexico, 2005.

[14] W. Zadrozny, M. Budzikowska, J. Chai, and N. Kambhatla, "Natural language dialogue for personalized interaction." in *Communications of the ACM*, vol. 43, 2000, pp. 116–120.

[15] Ernst and Young, "Global online retailing," in *An Ernst & Young special report*, 2000, pp. 85–86.

[16] P. Kourouthanasis, D. Spinellis, G. Roussos, and G. Giaglis, "Intelligent cokes and diapers: Mygrocer ubiquitous computing environment," in *In Proceedings of the 1st International Mobile Business Conference*, July 2002, pp. 150–172.

[17] S. Loke, "An exploration of agent assistance for physical marketplaces: Proximity-based reverse auctions," in *In Proceedings of the 2003 International Conference on Intelligent Agents, Web Technologies and Internet Commerce (IAWTIC 2003)*, 2003.

[18] M. Wagner, W.-T. Balke, and W. KieBling, "An xml-based multimedia middleware for mobile online auctions," in *Enterprise Information Systems III*. Netherlands: Kluwer Academic Publishers, 2002, pp. 259–269.

[19] A. Jagoe, "Mobile location services: The definitive guide." Prentice Hall, 2003.

[20] M. Attane and J. Papi, "E-parking: User-friendly ecommerce to optimize parking space," in *In Proceedings of M-Business*, 2002.

[21] B. Kolmel and S. Alexakis, "Location based advertising," in *In Proceedings of the 1st International Conference on Mobile Business*, Greece, July 2002.

[22] G. Chen and D. Kotz, "A survey of context -aware mobile computing research," Dartmouth Computer Science Technical Report TR2000-381, Tech. Rep., 2000.

[23] Wikipedia, "Wikipedia link for aiml, http://en.wikipedia.org/wiki/aiml," Tech. Rep.

[24] OpenNLP, "Opennlp online link, http://www.opennlp.org," Tech. Rep.

[25] Y. Matsuo and M. Ishizuka, "Keyword extraction from a single document using word co-occurrence statistical information," *International Journal on Artificial Intelligence Tools*, vol. 13(1), pp. 157–169, 2004.

[26] Microsoft, "Microsoft speech api (sapi), http://www.microsoft.com/speech/developers.aspx," Tech. Rep.

[27] wikipedia, "Likert scale, http://en.wikipedia.org/wiki/likertTech. Rep.