Context-Based Personalization for Mobile Web Search

Mario Arias
GRINBD. Univ. of Valladolid
47007 Valladolid, Spain
mario.arias@gmail.com

José M. Cantera
Telefónica I+D
47151 Boecillo, Spain
jmcf@tid.es

Jesús Vegas
GRINBD. Univ. of Valladolid
47007 Valladolid, Spain
jvegas@infor.uva.es

ABSTRACT

User experience while searching for web pages on the move can be far from satisfactory due to the inherent limitations of the input modes available in mobile devices. On the other hand, end-users can benefit from the availability of context-aware information anywhere, anytime. To overcome the usability problem and exploit context information at the same time, we propose a thesaurus-based *semantic context-aware autocompletion* mechanism. Our system can help the user in completing the desired query terms avoiding manual typing. In addition we are capable of filtering out non-relevant query terms for the Context in which the search process is conducted. Our context-aware proposal is based on a model which represents formally all the information about the user circumstances, the access mechanism (device and web browser) and the surrounding environment. Our evaluation reveals that users can find new relevant context-aware results with less effort.

1. INTRODUCTION

Mobile devices have evolved to provide bigger full-color screens, enhanced processing power and faster and permanent broadband Internet connections. These technologies have brought the World Wide Web to mobile devices introducing new requirements and expectations. Nonetheless, the vast majority of web sites and search engines are usually designed with desktop computers in mind. For that reason, current mobile search experience is far from satisfactory [17]. Search engine analysts, being aware of this problem, have designed mobile-oriented views to provide the same service from a smaller interface. Content transformation (reformatting) proxies have been devised to reduce pages on the fly [8], making them more accessible for the mobile web. Such approach deals with one of the major limitations of the mobile web, screen size, but it does not address the problem of the limited input modes. Furthermore, it does not take advantage of the contextual information available in a mobile environment in order to provide more accurate results.

Permission to copy without fee all or part of this material is granted provided that the copies are not made or distributed for direct commercial advantage, the VLDB copyright notice and the title of the publication and its date appear, and notice is given that copying is by permission of the Very Large Data Base Endowment. To copy otherwise, or to republish, to post on servers or to redistribute to lists, requires a fee and/or special permission from the publisher, ACM.
VLDB '08, August 24-30, 2008, Auckland, New Zealand
Copyright 2008 VLDB Endowment, ACM 000-0-00000-000-0/00/00.

Deficient handset's input modes are one of the most important limitations when using a mobile search engine. According to recent studies [9], users need an average of 40 key-presses with a 12-key keypad, and about 40 seconds to enter a query with a cell phone. That is a too big effort, users get easily tired and sometimes even give up. This leads to the *mismatched query problem* [5], mobile queries become even shorter and more ambiguous than the same queries in a desktop environment. The immediate consequence is that they do not reflect user's intentions precisely, hence the obtained results are poor. Ultimately the user is forced to visit a big number of useless web sites and to navigate through several result sets in order to find the needed information. This search flow might be acceptable when surfing the web from a desktop computer, but it becomes an extremely arduous task when performed from a mobile device.

Mobile Web Search introduces new challenges not present in traditional web search. Users normally own modern cell phones which allows them to be permanently online anywhere, anytime. A typical mobile web search scenario consists of a user outdoors with an information need. At this point he takes his phone and uses a web search engine to find an answer to a query. Furthermore, he is probably doing something else at the same time, like walking or talking to a friend. In such situation the user needs a short, fast but also accurate answer to his query.

Autocompletion mechanisms appear as a natural improvement to the limited input modes problem. In fact, any modern information retrieval system can obtain the most popular query terms or phrases from indexed pages. Such elements can be used to interactively complete user's queries. An autocompletion engine can be helpful both by saving typing time and by finding new, serendipitous terms. However a pure syntactic approach can present difficulties, because it will be based on the coincidence of string representations (words), but not in concepts and their relationships. The corollary is that, the autocompletion mechanism will be helpful if the user intends to use the same word that the system is expecting, but if, for instance, a synonym is in user's mind, the system will be unable to recognize and autocomplete it. The previous facts have led us to the introduction of a concept-driven *semantic and context-aware autocompletion engine.*

Initially, we have created a thesaurus which represents concepts together with their synonyms and relationships. We have modeled concepts corresponding to the domain of services offered to the public, such as hotels, restaurants, tourism offices, public transport stations and similar. We

have chosen such set of concepts, as they are good candidates to be the most likely-to-be-used query terms in a mobile environment (where the user is always looking for something to address a very specific and punctual necessity). For example, our thesaurus includes the concept "dinner" related to "food", "restaurant" and "supermarket".

An autocompletion mechanism targeted to the mobile environment can also benefit from the availability of contextual information. For example, there is no point in suggesting the term "beach" while in a place which it is not on the seaside, or during winter time. To deal with those scenarios, we have extended our semantic autocompletion engine with context-aware recommendation, which filters out non-contextually-relevant concepts. As a result we are able to suggest the best query terms according to each situation. To implement such advanced features we have introduced a formal Context Model which represents all the significative properties about the environment (place, access mechanism, user profile, etc.) in which the search process is conducted.

We have integrated our semantic and context-aware autocompletion system into a working prototype. Such prototype also includes a novel user interface designed to meet all the requirements imposed by the mobile environment. Finally, we have tested the feasibility of the system by making a qualitative analysis of the improvement obtained in the user experience.

The rest of this paper is organized as follows. Section 2 briefly describes the background and related work regarding context awareness, personalization and recommender systems. The detailed description of our proposal can be found on section 3. The evaluation and experimentation results are detailed on section 4. Finally section 5, is devoted to summarize the conclusions and the guidelines for future work.

2. BACKGROUND AND RELATED WORK

The expansion of embedded and handheld devices has promoted the research on the benefit of contextual information to improve Human-Computer Interaction, Ubiquitous Computing and Web Search experience [3].

A.K.Dey [2] suggests the following general definition of Context and context-awareness: *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 application themselves. A system is context-aware if it uses the Context to provide relevant information and/or services to the user, where relevancy depends on the users task.* According to this definition, the Context could be considered the current data displayed on the screen, the surrounding environment or even the whole application. For that reason every context-aware application must explicitly specify what information is part of the Context. Additionally, the Context is inherently dynamic and constantly changing. A few properties will be barely modified over time, like user name or age, but other ones may frequently vary, like time or position. We must ensure that when context properties are queried they are all up to date, or at least marked as expired [18].

The main challenge introduced by context-awareness is to come up with a flexible and unambiguous representation of the Context, including a framework to ease the development of context-aware applications. Based on this Context Model, applications can figure out which actions should be triggered or what data is potentially relevant to the user. Nonetheless, the heterogeneous nature of context-aware applications makes it impossible to have a universal, unique representation of the Context. However, a good compromise can be achieved if context models are able to manage a set of universal properties, useful for any application, in conjunction with application-specific, custom properties. Previous works [18] propose a Context taxonomy and a framework which define several abstract layers of knowledge. Such layers are ready to be mapped with the properties that actually will model the Context. In addition, formal reasoning techniques can be employed to create derived properties based on those which are directly fetched.

Another important issue has to do with gathering all the significative context information that can be of interest to an application. It is noteworthy that each application will be only interested in a limited subset of the Context. As a result, context frameworks should provide mechanisms to allow applications to express their interest in certain context properties. Additionally the context framework should be prepared to create bindings between context properties and the corresponding information sources. Such bindings should hide applications from the lower-level protocols or services used to actually obtain the context property values. Besides, some directly measured context properties are not suitable to use *as-is*; there can be properties that need a previous transformation or processing to be useful for an application. For example, an application might require GPS coordinates as input in some cases, but in others it might need the place name. As a consequence, a context framework should perform automatic context data transformations on behalf of the application.

To know which are the interests of each user it is needed to generate a user profile. The first option is to explicitly ask what are user's interests with a small survey form. This method is known to be *high quality* if the survey is correctly designed, but in general users dislike filling forms, especially when the benefits are not immediately obvious. In addition, users do not feel safe by submitting personal information to untrusted servers [11].

Collaborative filtering techniques analyze user behavior to create implicit user profiles. They take into account which links are more frequently clicked or the time spent on each one to extrapolate user interests without them even noticing. Users with similar interests can also be grouped into classes. Once the individual has been classified, assumptions about his interests can be made based on those of the whole group. This introduces the concept of social profiling, which can be valuable when a new user connects to the application for the first time and fewer details about him are available.

Commercial search engines, such as Google or Yahoo, incorporate context-independent autocompletion mechanisms intended to work in the desktop environment. Web based recommender systems have been successfully applied to modern online shopping sites, filtering and suggesting between a huge amount of available products [16]. Wietsma [20] developed a PDA-based prototype of a recommender system. Other works [10] have applied recommender systems to simplify mobile web search by offering syntactic query suggestions. However, none of them has combined semantic autocompletion mechanisms with context-aware recommendation in a mobile environment, which it is the main contribution of our research.

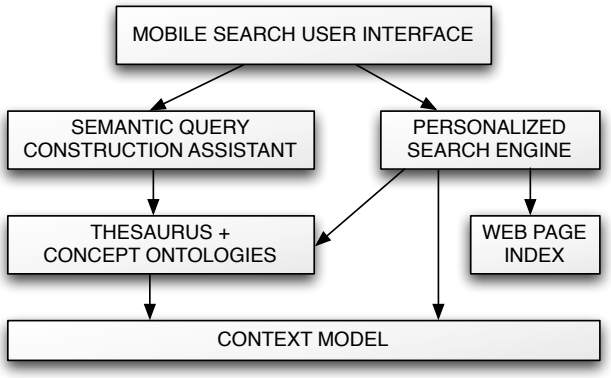

Figure 1: Mobile Web Search Framework.

Improving web search by exploiting contextual information has been previously studied, although there is no agreement on context interpretation and scope. There are authors that consider the history of visited pages as web search context. Following that criteria, they try to analyze and extract the most important keywords found in those pages, and use them to ensure that further queries will keep on subject [12, 4]. While these solutions can improve web search on the desktop, mobile search should go beyond, considering additional variables like location, conditions of the environment, points of interest nearby, weather situation or guessed user intentions [5].

Query expansion and substitution is a technique that can be used to include context information during the search process. The mechanism works as follows: Just before launching the query it is completed with additional terms [5, 13]. Such terms can be synonyms, disambiguating subject-related words or special keywords aimed at clarifying the intention, like *how to*, *ways to*, *what is*. Another way to contextualize search is by re-ranking the obtained result set according to certain properties: subject, proximity to user's location or intentions.

The use of the above methods will ensure that the most contextually-relevant results will appear on top of the result set, thus reducing reducing the number of interactions needed to find proper results [19, 11].

3. OUR PROPOSAL

To overcome previously described limitations, we have designed a framework to combine different paradigms and novel ideas from several sources, separated in different modules which work together. In first place we need to gather as much information as possible from user and environment. Then, based on that information, a recommender system selects the most relevant suggestions for user guidance. We have identified the following modules (See figure 1):

- A **Context Model** which provides a formal representation of the user, the environment and the access mechanism, enabling personalization and context-awareness.
- A **Thesaurus**, which models semantic relationships (synonym, broader, narrower, related) among the specific concepts managed by the search system.
- An **Ontology** which describes additional properties that each thesaurus concept may have. They are used to offer additional options to let the user choose instead of type.
- A **Semantic Query Construction Assistant** intended to recommend best-suitable options in context to avoid users from typing, and thus personalizing the search process.
- A **Personalized Search Engine** which makes use of all information available (the query, selected thesaurus concept, ontology values) to find relevant results. To obtains results it makes use of a traditional web search engine which maintains a web page index.
- A **User Interface** prototype adapted to the peculiarities of mobile devices, which makes use of all previous modules to provide an easier user interaction.

Our system is capable of suggesting search terms based on concepts of the thesaurus instead of words. The outcome is a more abstract level of recommendation as the system adapts to the user ideas, decoupling the system from any specific vocabulary. This solution ameliorates traditional approaches, which are restricted to syntax comparisons. Furthermore, we filter out concepts that are unlikely to be chosen in that specific concept, so less options are recommended, just the most relevant ones. In addition, each thesaurus concept is complemented with an ontology describing additional properties of that idea. Once the user have selected a specific concept, additional options are automatically presented so he just have to choose to refine his query instead of typing. For instance, if the user has selected the concept "Restaurant", a form with two fields "price" and "type of cuisine" will automatically appear.

3.1 Context Model for Personalization

The foundation for a context-aware application is a formal context specification which maintains all properties and provides a standard access to them. We used an ontology to describe all entities of the domain together in a taxonomy. We have decided to use the OWL language [15] as it is a widely accepted industry standard and therefore there are several open source tools (parsers, reasoners, editors, third-party ontologies) available to ease development.

We propose an extensible Context Model divided into three layers (Figure 2):

- **Directly Fetched Properties**. These are properties that can be automatically gathered from context information sources. For example the location coordinates obtained from a GPS sensor or the born year directly specified by the user.
- **Derived Properties**. These are "implicit" properties that can be inferred or calculated from other properties. They constitute a higher abstraction layer on top of the directly fetched properties, so they are more easily comparable against application-level ideas. For example, taking into account the location coordinates provided by a GPS, the system can obtain the name of the country and region from a GIS service, or it can calculate the user's age based on his born year.

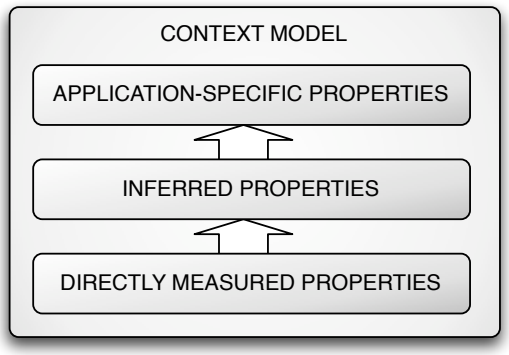

Figure 2: Context Layered Model.

- **Application-Specific Properties**. Applications might want to add or redefine properties by using their own rules, in order to extend the base context definition to better fit concrete application needs. For example, an application can define a property called *nice day* given the temperature and the atmospheric condition. All these definitions are identified at application design time, but at some point it may be required to differentiate what is a nice day according to the country. This operation can be easily accomplished by just modifying the corresponding rule.

The SWRL [6] language has been used for rule definition as it is the natural extension to OWL when rule-based depictions are needed. SWRL provides syntax and semantics to define rules affecting OWL classes, properties and individuals.

For example, we can define the following rule to create the abstract property called *niceday*:

$$weather(?w) \wedge temperature(?w, ?t) \wedge greater(?t, 20) \wedge sunny(?w, true) \rightarrow niceday(?w, true)$$

where $?w$ refers to a OWL individual which belongs to the *weather* OWL class, *temperature* is a property which links a weather instance to its temperature, *greater* is a SWRL built-in function, *sunny* is a boolean property, and *niceday* is the new defined property.

Once the context framework is defined, we proceed to identify the relevant classes, properties and relationships for any context-aware mobile web search application:

- *User Profile* This category contains all the implicit and explicit properties related to the user and his circumstances. We model several properties such as the preferred language, the date of birth, the place where he lives, etc. A FOAF [1] extension is used for user profile modeling.
- *Device and Browser* This set of properties describe the characteristics of the user's device and web browser. Such properties make it possible to distinguish between a cell phone and a PDA, or between a rich AJAX-enabled browser and a first generation limited browser. Detailed information is crucial in order to take full advantage of hardware capabilities, for example devices with bigger screens can show more rows of a table at the same time than those with smaller ones.

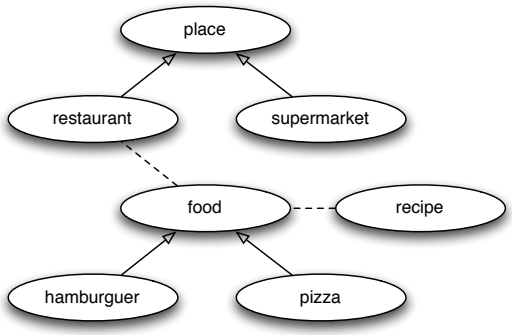

Figure 3: Thesaurus concept lattice example.

- *Geospatial Context* We model the current user location not only in terms of geographic coordinates but also in terms of the kind of place where the user is (on the seaside, on the countryside, at an airport or station, etc.). Such contextual information can be later used to promote the tasks which are more typical in those certain situations.
- *Environment Conditions* These are properties that model the surrounding environment. In our prototype we model mainly the weather conditions gathered from Web Services published on the Internet. For example, if the user is searching for leisure activities and the weather is rainy, our recommender system should suggest theater or cinema, rather than outdoor activities.
- *Date and Time* They are useful to restrict behavior at certain times or dates. For instance, there is no point in going to the cinema at eight o'clock in the morning. In addition it can be used to trigger certain actions when certain events occur.

3.2 Semantic Context-Aware Recommendations

We propose a context-aware thesaurus-based recommender system as a natural approach to semantically guide the query construction process.

In first place, we need a domain-specific thesaurus for recommendations, which must cover all relevant concepts of the domain and their relationships. We distinguish among three kind of relationships: broader, narrower and related. For example, "place" is a broader term for "restaurant" and "museum"; and "food" is related to "restaurant". Each concept also includes synonyms that may refer to the same idea, for example the concept "soccer" will also include the synonym "football". Moreover, these words can be specified in different languages without altering thesaurus structure, thus simplifying internationalization.

For our working prototype, we have tailored a thesaurus that covers concepts in the domain of transport, leisure and public services. In our opinion, they are the more likely to be useful in a generic-purpose mobile web search engine. Our thesaurus implementation has been defined in RDF [14] using the SKOS [7] vocabulary. It contains 100 concepts and about 200 relationships. As an example, figure 3 depicts one branch of our thesaurus.

The recommendation scenario works as follows. Once the user has typed one or more characters, the recommender system asks the thesaurus for concepts which start with those

Figure 4: Thesaurus-based suggestions tree.

Figure 5: Thesaurus concept and its ontological properties.

letters, of course including synonyms. Then it applies a rule-based filter to discard concepts which are not likely to be useful in that situation and to promote concepts that seem more interesting. The remaining concept list is sorted using a score system based on previous selections, and the best ones are selected to be shown. Instead of showing just the word for each concept, we also include its broader, related and narrower concepts in the same line, arranged in a tree view (as depicted in Figure 4). Finally, the user can select whichever option he deems more appropriate to fulfill his needs.

The first major feature of our proposal is context-awareness. Based on our context definition, we specified which are the best conditions for each thesaurus concept to be useful, and in which cases it could be directly discarded. These conditions are set by using SWRL rules, because the Context definition and the thesaurus can be joined together, and be seen as a unique ontology. This filter considerably reduces the number of options, whereas the remaining ones are the most interesting for a specific context.

A typical thesaurus rule matches the following pattern:

$$concept(?c) \wedge context(?t) \wedge \{conditions(?t)\} \rightarrow isSuitableInContext(?c, ?t)$$

Then, the *isSuitableInContext* property specifies which concepts are considered adequate in that specific context instance. Likewise, a property called *isNotSuitableInContext* asserts which concepts will probably be useless and should be discarded. The knowledge engineer is in charge of identifying under what conditions concepts are favorable or undesirable, based on his expert experience.

Our solution can also be used to solve sense ambiguity within queries. A common example of sense ambiguity is the word *jaguar*, which could refer to the animal, the car manufacturer or the Apple Operating System among others. Each of the senses has its own thesaurus concept, as they are homographs but represent completely different ideas. When there are two or more ambiguous options to recommend, the system offers all of them in separate lines, each one with its own relationships. In this case, the broader is really important to let the user distinguish every single meaning. For the *jaguar* example, the word will appear three times, one with *animal* as broader, another with *Car Manufacturer* and the last one with *Operating System*. When the user selects one, the system knows univocally to which one he refers in the thesaurus, and annotates the concept identifier to later expand the search engine query with disambiguating terms.

The last step of data acquisition is the use of the ontology to gather more information about user intentions and requirements. The ontology is designed to provide additional properties of the thesaurus concepts, together with their common values. Once the user selects a single concept, the system constructs a form to ask the user for more details, as shown in Figure 5. This can be used to construct a more concrete query which reduces ambiguity, but it is also a way of explicit user profiling, as the user is who specifies which options he prefer. In this case the user will be prompted to fill a small and query-specific form which takes just a few seconds and causes a direct effect on results. This beats other explicit solutions, which require users to fill big, generic, boring and time-consuming forms.

3.3 Personalized Search Engine

The semantic query construction assistant module is able to simplify user input task, but it also gathers a fair amount of extra information. We need to describe how to convert this information to a suitable format that will be processed on top of a traditional web search search engine.

There are two simple ways to employ all gathered information to affect search results. The first one is doing query expansion, given the selected thesaurus concept and the fields from the ontology, we can construct a new query by adding new terms to the user-typed query before accessing the traditional search engine. Another option is analyzing provided results after they are returned by the search engine, to filter out those which do not match user intentions. In this work we focus on the first one, query expansion.

The first consideration to take is that the user has a text field to type his own query in a traditional manner. Of course the system must be able to provide results even if any thesaurus concept or ontology option was selected. Therefore, the user query will have higher priority over the rest of the gathered information.

If the user selected one thesaurus concept we can profit that information to drive the search. For example we can add its synonyms and related concepts as optional terms, so web pages matching those subjects will gain weight in the search engine scoring system. This approach is interesting when a huge number of web pages is available, because it provides more specific results. On the contrary, when the amount of provided results is too scarce, we can try to launch a secondary query by substituting the main concept by one of the related ones. Of course this will result in a

performance and quality decrease, but this option is better than wasting user time and bandwidth usage to show a *No results found* webpage.

3.4 Enhanced User Interface

The heterogeneity of Mobile devices and browsers have led us to build a user interface that adapts to the capabilities of the target delivery context. For latest mobile browsers and devices, such as Windows Mobile PDAs or Apple's iPhone, we based it on HTML 4, AJAX and Javascript. For conventional mobile phones it has been degraded gracefully to XHTML-MP, providing a minimal but at the same time functional user experience. In every case, we tried to take full advantage of the screen and to minimize de number of clicks needed to navigate through the interface.

For enhanced web browsers we propose a tab-based search interface (see Figure 5): the first tab contains query terms and suggestions, the second one search results, and the last one allows the user to browse and modify some of the context properties, like current location, language, or situation (at work, at home, on a trip). When any of the search conditions change, the results tab is automatically updated to reflect those changes.

The search query tab contains an input box where the user can type his query in the traditional way. We wanted to maintain the traditional interaction flow to prevent the user from feeling lost. However, while he is typing, all the concepts (and related ones) that match the entered letters will appear on a tree view. Then he can select any of the suggested concepts (coming from the thesaurus), and finally additional options like concept properties (coming from the ontology) will appear (see Figure 5).

With regard to the implementation, it is important to note that we have taken advantage of the AJAX capabilities present in enhanced devices. For example, we minimize the traffic between the device and the server, making the user interface more responsive. We have also developed a browser plugin for the Windows Mobile Platform which provides access to Context information only accessible from the device, such as the GPS coordinates. Once they are obtained, the client properties are sent to the server, and they will be incorporated to the context definition as any other property.

4. EVALUATION

We invited 12 people, divided in 3 groups, to evaluate our prototype. The group A was composed by students in the first year of MsC in Computer Science; the second group, B, was integrated by IT technicians, and the third group, C, users not familiar with the mobile environment. We let them interact with our system using a real mobile device for a few minutes until they got used to it. Then we asked them to search for different kinds of information, including both subjects present and absent in the thesaurus. Finally they fulfilled a small survey containing questions regarding their user experience, particularly about the usefulness of the autocompletion system.

The most important point we wanted to test by using this evaluation was the level of satisfaction of final users. All members of our test groups found our suggestion system useful and intuitive, and they were able to use the system without trouble. They had the chance to directly compare the needed effort to type queries with our recommender system and without any aid tool. They agreed that the semantic recommender system reduces typing time and allows to use the system easier.

Less skilled people did not expect suggestions to appear, so they did not realize at the first time about the autocompletion feature. For that reason, we insist in the importance of maintaining the traditional search flow unmodified (type query, click on search button, browse results), while at the same adding the new features. When those users finally discovered the power of autocompletion they were the most amazed, as less skilled people normally needs even more time to accomplish a task in the absence of a recommendation system.

We also wanted to know their opinion about the multi-tab design applied to mobile web search, which permits to switch between the query/autocompletion page and the results page. They constructed the query with the autocompletion system, and then browsed the result page. When the results were not good enough to satisfy their information need, they naturally returned to the autocompletion tab to refine the query. They found easier to select another thesaurus concept or edit any of the ontology options rather than typing a new query from scratch. Indeed, users typically need several refinement iterations to reach their results, so our system let the users save even more time.

Our experiments revealed that the response time was longer than expected. In fact users got nervous and started switching tabs before waiting to load which resulted in even longer waits. For that reason we reduced load time by saving a local copy of each tab on the client side, minimizing the interactions with the server; now when the user is switching between tabs, and no option has changed, the cached content is used, therefore saving bandwidth and time.

5. CONCLUSIONS AND FUTURE WORK

We have described the limitations experienced by mobile web search users, focusing on lack of personalization and unconvenient input modes. We have developed a *semantic context-aware autocompletion* mechanism, based on a layered Context Model, a thesaurus to represent the subject domain, and an enhanced user interface. Our results show that these techniques provide a richer search experience.

We based our work on existing ideas from diverse research areas, like contextual applications, recommender systems and information retrieval query expansion. Then we studied how to join all of them together and construct a working prototype, which serves as proof of concept for the feasibility of the integration.

We have highlighted the importance of context modeling as the basis to provide personalization within mobile web search. Our proposed Context Model meets all the requirements imposed by a mobile web search environment. This is a first step towards a personalized access to the vast content of the World Wide Web. We proposed SWRL rules in order to extend de powerfulness of the context definition, allowing knowledge engineers to capture real world facts in a explicit manner.

We have observed that the thesaurus is a useful tool for several steps in the search process. It serves as guide for the recommender system providing the concepts of the domain, it is also useful to disambiguate user intentions and it can be easily annotated with rules to distinguish under what conditions each concept will be more or less interesting.

Our prototype is currently in an early phase and there is still a lot of research to do. We are working on integrating advanced user profiling algorithms to characterize user intentions in a more precise way, because knowledge about user is a fundamental piece in improving the relevancy of the results obtained during the recommendation process. We are also working on the personalized search module, analyzing different options of query expansion and result scoring to know how they improve search results.

6. ACKNOWLEDGMENTS

This work has been partially supported by TIN2006-15071-C03-02 project, Ministry of Education and Science, and by the Government of the region of Castilla y León through the Agency for Economic Development, (ADE), Spain.

7. ADDITIONAL AUTHORS

Additional authors are: Pablo de la Fuente (GRINBD, email: `pfuente@infor.uva.es`), Jorge Cabrero (GRINBD, email: `reybamba@gmail.com`), Guido García (ITDeusto, email: `ggarciab@itdeusto.com`), César Llamas (GRINBD, email: `cllamas@infor.uva.es`), and Álvaro Zubizarreta (GRINBD, email: `zubisoft@gmail.com`.

8. REFERENCES

[1] D. Brickley and L. Miller. FOAF vocabulary specification. Technical report, FOAF, nov 2007. http://xmlns.com/foaf/spec/.

[2] A. K. Dey. *Providing architectural support for building context-aware applications*. PhD thesis, Georgia Institute of Technology, Atlanta, GA, USA, 2000. Director-Gregory D. Abowd.

[3] P. Dourish. Seeking a foundation for context-aware computing. *Human-Computer Interaction*, 16(2/4):229–241, 2001.

[4] L. Finkelstein, E. Gabrilovich, Y. Matias, E. Rivlin, Z. Solan, G. Wolfman, and E. Ruppin. Placing search in context: the concept revisited. In *WWW '01: Proceedings of the 10th international conference on World Wide Web*, pages 406–414, New York, NY, USA, 2001. ACM.

[5] S. Hattori, T. Tezuka, and K. Tanaka. Context-aware query refinement for mobile web search. In *SAINT-W '07: Proceedings of the 2007 International Symposium on Applications and the Internet Workshops*, page 15, Washington, DC, USA, 2007. IEEE Computer Society.

[6] I. Horrocks, P. F. Patel-Schneider, H. Boley, S. Tabet, B. Grosof, and M. Dean. SWRL: A semantic web rule language combining OWL and RuleML. W3C member submission, W3C, may 2004. http://www.w3.org/Submission/SWRL/.

[7] A. Isaac and E. Summers. SKOS simple knowledge organization system primer. W3C working draft, W3C, feb 2008. http://www.w3.org/TR/2008/WD-skos-primer-20080221/.

[8] Jo Rabin and Andrew Swainston. Content Transformation Landscape 1.0. W3C Working Draft, W3C, Oct. 2007. http://www.w3.org/TR/2007/WD-ct-landscape-20071025/.

[9] M. Kamvar and S. Baluja. Deciphering trends in mobile search. *Computer*, 40(8):58–62, 2007.

[10] M. Kamvar and S. Baluja. Query suggestions for mobile search: understanding usage patterns. In *CHI '08: Proceeding of the twenty-sixth annual SIGCHI conference on Human factors in computing systems*, pages 1013–1016, New York, NY, USA, 2008. ACM.

[11] K. Keenoy and M. Levene. Personalisation of web search. In *Intelligent Techniques for Web Personalization, IJCAI 2003 Workshop, ITWP 2003, Acapulco, Mexico*, pages 201–228, 2003.

[12] R. Kraft, F. Maghoul, and C. C. Chang. Y!q: contextual search at the point of inspiration. In *CIKM '05: Proceedings of the 14th ACM international conference on Information and knowledge management*, pages 816–823, New York, NY, USA, 2005. ACM.

[13] S. Lawrence. Context in web search. *IEEE Data Engineering Bulletin*, 23(3):25–32, 2000.

[14] F. Manola and E. Miller. RDF primer. W3C recommendation, W3C, feb 2004. http://www.w3.org/TR/2004/REC-rdf-primer-20040210/.

[15] D. L. McGuinness and F. van Harmelen. OWL web ontology language overview. W3C recommendation, W3C, feb 2004. http://www.w3.org/TR/2004/REC-owl-features-20040210/.

[16] Q. N. Nguyen and F. Ricci. User preferences initialization and integration in critique-based mobile recommender systems. In *Artificial Intelligence in Mobile Systems 2004, in conjunction with UbiComp 2004*, pages 71–78, Nottingham, UK, 2004.

[17] V. Roto, A. Popescu, A. Koivisto, and E. Vartiainen. Minimap: a web page visualization method for mobile phones. In *CHI '06: Proceedings of the SIGCHI conference on Human Factors in computing systems*, pages 35–44, New York, NY, USA, 2006. ACM.

[18] A. Schmidt. A layered model for user context management with controlled aging and imperfection handling. In *Modeling and Retrieval of Context, Second International Workshop, MRC 2005, Edinburgh, UK*, pages 86–100, 2005.

[19] A. Sieg, B. Mobasher, and R. Burke. Ontological user profiles for representing context in web search. In *WI-IATW '07: Proceedings of the 2007 IEEE/WIC/ACM International Conferences on Web Intelligence and Intelligent Agent Technology - Workshops*, pages 91–94, Washington, DC, USA, 2007. IEEE Computer Society.

[20] R. T. A. Wietsma and F. Ricci. Product reviews in mobile decision aid systems. In E. Rukzio, J. Hkkil, M. Spasojevic, J. Mntyjrvi, and N. Ravi, editors, *PERMID*, pages 15–18. LMU Munich, 2005.