AN ADAPTATION MANAGER FOR PERSONALIZED MOBILE WEB SEARCHES

Wolfgang Woerndl and Huda Yousef
Dep. of Computer Science, Technische Universitaet Muenchen, Boltzmannstr. 3, 85748 Garching, Germany

Keywords: Mobile Web, Personalization, User Profile, Context, Context-aware Applications, Web Searches.

Abstract: Context-aware and personalized information access becomes more and more important, especially in a mobile scenario. In this paper, we present an Adaptation Manager which customizes Web searches to a user's context and profile attributes. The approach allows for reuse of personal information for different services such as the Google Web Service API. We identify personalization components as parts of our Adaptation Manager and explain the design of the system. We have also implemented the most important parts of the system and thereby show the usability of the approach. Our solution is extensible, for example to incorporate a rule system to (semi-)automatically switch between services.

1 INTRODUCTION

New developments in technology along with the growth of mobile and wireless communication provide us with services and information anytime and anywhere. However, the access of relevant information is not easier but even more complicated than before. Users are suffering from too much information and have trouble finding relevant answers to their queries. The omnipresent information overload makes an impact especially with mobile Web searching and browsing:

- Users cannot browse through many search results with the small screen of a mobile device in contrast to working on a desktop PC. It would be more desirable to pre-filter results to adapt to the user's current needs, if possible.
- When using mobile devices, e.g. when travelling, context based information is more important. Users need access to restaurants or other points of interests in their vicinity, as an example for context.

So while on the one hand personalized and context-adapted information is more important in a mobile scenario, on the other hand obtaining relevant information is also more difficult. This is in large part due to the restrictions regarding display size, bandwidth and input capabilities of mobile devices.

In our scenario, a travelling user uses a mobile device, e.g. pocket PC or smartphone to access services such as the Google Web search. The user wants to retrieve context-aware personalized information, for example an answer to the query: "which italian restaurants are in the vicinity of my current location that fall in my price range". In other words, users want customized information and not have to manually sort out Web search results. To do so, our solution proposes a server-based personalization system based on the current context (e.g. location) of the user and information that has been collected about her (the user profile).

The main goal of this paper is to describe the "Adaptation Manager" which is the core of the approach. Thereby, we first explain our general architecture and approach (section 2). Then, we will explain our Adaptation Manager in more detail and also provide information about some design issues. We also present our implementation and the user interface. In Section 4, we will outline some of the research issues we work on to further improve the system. Finally, our paper concludes with a brief summary and discussion of related work.

Woerndl W. and Yousef H. (2007).
AN ADAPTATION MANAGER FOR PERSONALIZED MOBILE WEB SEARCHES.
In *Proceedings of the Third International Conference on Web Information Systems and Technologies - Web Interfaces and Applications*, pages 156-161
DOI: 10.5220/0001268401560161
Copyright © SciTePress

2 THE BIG PICTURE

To fulfil the vision that is outlined in the introduction, we propose the architecture that is depict in Fig. 1. Thereby, users do not access information services such as Google directly, but through an Adaptation Manager. This component customizes information retrieval and adapt to the current device of the user. The Adaptation Manager queries other services to satisfy user requests. In our approach, a ”Service” is any software that can be used in to access information, for example Google Web search or a tourist information guide. Helper applications are used to attain context information, e.g. retrieve address information from GPS coordinates (see chapter 3.2.2).

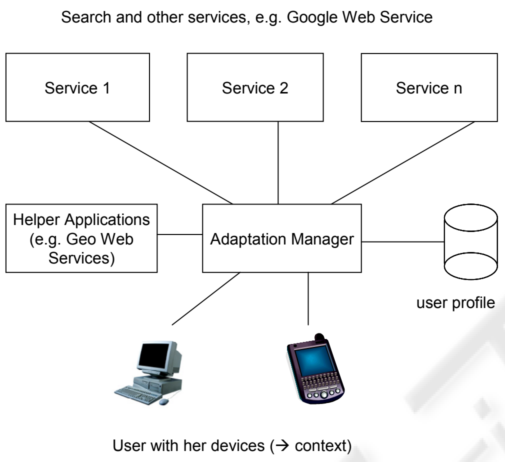

Search and other services, e.g. Google Web Service
Service 1
Service 2
Service n
Helper Applications (e.g. Geo Web Services)
Adaptation Manager
user profile
User with her devices (→ context)

Figure 1: Architecture overview.

We distinguish between “context” and “profile” in our system. We use the context definition according to Dey et.al.: “Context is any information that can be used to characterize the situation of entities (i.e. whether a person, place or subject) that are considered relevant to the interaction between a user and an application, including the user and the application themselves” (Dey, Abowd and Salber, 2001). In our approach, context is very transient (e.g. the location of a user at a given time) and not stored permanently. On the other hand, we define “(user) profile” as information about a user that is managed on a server to personalize services. Examples include emails addresses and demographic information about the user, and also interests or preferences, e.g. the user’s favourite restaurant type etc. Both context and user profiles are used to personalize information access.

We propose a "thin client" approach, that means the actual personalization is done in the Adaptation Manager on a server, not on the end user device. One main advantage is that user profile information can be reused for different services, even when using different devices. The user can also modify her profile using a desktop PC and apply these configurations to adapt information access on a PDA later, for example.

Therefore, in our approach only the context acquisition is done on the client. This includes retrieving GPS coordinates on a GPS-enabled pocket PC, for example. The actual context acquisition is out the main focus of this paper, we will concentrate on the Adaptation Manager, which will be explained in the next section.

3 THE ADAPTATION MANAGER

3.1 Requirements and Subcomponents

We have identified the following main requirements for our system and the Adaptation Manager:

- Easily usable from a user’s perspective, even on PDA’s and smartphones
- Accepting and analyzing user queries (keyword searches)
- Management of user profiles in a database and enhancing queries with profile attributes
- Interpreting and using context data that is transmitted by the mobile end user device, in particular GPS coordinates
- Accessing other (Web) services to fulfil user queries
- Evaluation of search results and adaptation to the user device
- Modular and extensible software design

As parts of our adaptation manager, we designed two main components: `QueryProcessing` (including a client to the search – and other – services) and `ResultEvaluation`. The `QueryProcessing` handles the user search requests and takes case of the personalization. The `ResultEvaluation` adapts to the device of the user.

3.2 Design

One of the main design goals of our approach was to develop the individual subcomponents as independent from each other as possible using software engineering methods (Brügge and Dutoit, 2004). Therefore, the system can be easily extended with new features or services.

3.2.1 Program Flow

The basic program flow in our Adaptation Manager is as follows:

1. `QueryProcessing`: Evaluate context, in particular transform GPS coordinates to addresses (see chapter 3.2.2), and figure out the device the user is using
2. `QueryProcessing`: Choose an appropriate service (see 3.2.3)
3. `QueryProcessing`: Actual personalization, e.g. add user profile attributes to improve searches (see 3.2.4)
4. `ServiceClient`: Propagate enhanced and personalized request to service, e.g. Google Web Service to perform a (Web) search
5. `ResultEvaluation`: Filter search results
6. `ResultEvaluation`: Adapt layout of the result Web page to the actual end user device (see 3.2.5)

We explain some selected aspects in more detail in the following subsections.

3.2.2 Geocoding Services

Since location-based services are the most prominent example of context-aware applications, we explain in this subchapter how to deal with location information. We assume that GPS coordinates are provided by the end user device and propagated to our Server application. While most mobile phones are not equipped with GPS right now, it is safe to assume that more and more devices will integrate a GPS receiver in the future.

Especially interesting for our purposes are "reverse geocoding" services. These services transform GPS coordinates into actual addresses (street and location names etc.) and can be used to find businesses at a certain location. Reverse geocoding services can be either database applications or Web Services. Existing applications include OpenGeoDB (http://opengeodb.hoppe-media.com/index.php?FrontPage), Mapbender (http://sourceforge.net/project/showfiles.php?group_id=88554) or the Geonames (http://www.geonames.org) databases. More flexible are Web Services, for example provided by ViaMichelin (http://ws.viamichelin.com/) or Geonames. The latter offers "find nearby postal codes" and "find nearby place names" queries based on GPS coordinates. We have integrated the Geonames Web Service to retrieve post codes and place names in our implementation.

The geocoding services are an example of a helper application to improve search requests in our architecture; it is also possible to access map services and generate maps for the user. While applications exist that provide location-based solutions, we handle context separately from services to allow for a flexible and extensible architecture.

Apart from location, other context attributes include the current date and time which could be connected with a calendar of the user. We are currently working on generating high level context from sensor data, to figure out what the user is doing in which environment with her (mobile) device right now.

3.2.3 Integrating Services

As an example of the design of the Adaptation Manager, we show how the connection of (search) services in the `QueryProcessing` component is designed (Fig. 2). The software design pattern that is used here is the "strategy pattern" (Brügge and Dutoit, 2004). This pattern allows for interchangeable services. Thereby, the services are decoupled from the rule system that performs the selection of services.

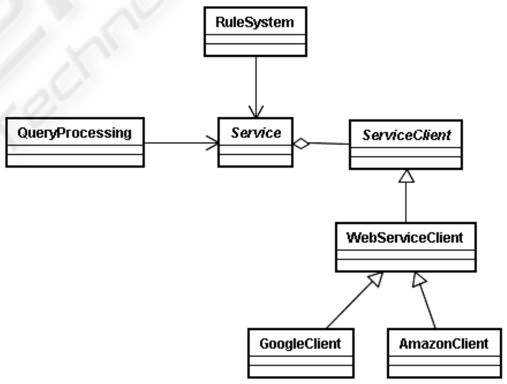


Figure 2: Design QueryProcessing.

The (abstract) `Service` class encapsulate search and other information services. The `RuleSystem` class is responsible for selecting the service that is used for a query. Possible services which can be integrated include (local) database applications, Web Services or RSS feeds, for example for analyzing Weblogs (Woerndl, Groh and Toni, 2006). The `GoogleClient` implements the actual Client for the Google Web Service API, for example. Using this design, new services can easily be integrated and the system can be gradually extended.

3.2.4 User Profiles and Personalization

Our user profile consists of attribute-values pairs, e.g. “restaurant.type = Italian”. The personalization is done by extending the keyword-based search string with some information from the profile and also by filtering the search results afterwards.

At this time, we use a simple rule system that maps user queries to user profile attributes, for example: <<if the query contains the string “restaurant” then add the value of “restaurant.type” from the profile to the query>>. The user profile information could be reused for other relevant services, e.g. when accessing a purpose-built restaurant data base.

However, not all elements of the personalization are possible reusable because of different capabilities and options of services. Service-dependent personalization has to be configured and possibly implemented for each service. For example, the Google Web Service API offers an option to specify the languages of search results, so the `QueryProcessing` retrieves a configuration setting “language” from the profile to personalize the search request.

By keeping profile information apart from the services, users can enter and manage information about themselves with a Web based interface from their desktop PC and retain the control about their personal information (privacy). For example, if the user’s favourite restaurant type changes, she can easily change this setting once for usage with different service providers. It is also possible to integrate a component for implicit user modelling in the Adaptation Manager, but in any case the user has always access to what information is collected about her and can delete or modify unwanted user profile attributes.

3.2.5 Customizing for User Devices

The main options for mobile applications are to either generate HTML or WAP pages, or develop Client-side J2ME (Java Mobile Edition) or Windows Mobile (resp. “Compact Framework”) applications. We have designed our Adaptation Manager to generate XHTML pages, since Web browsers are provided for most PDA’s and similar devices.

To adapt the layout of the XHTML pages, we keep information about device capabilities in the Adaptation Manager. This is part of the user profile, therefore users can configure themselves, how many results are displayed on which device (or use the default configuration for a certain device), for example.

We also propose a template based mechanism to customize the appearance of search result pages. A template is provided for each (type of) device and is used to create the actual XHTML page that is send to the Client.

3.3 Implementation

We have implemented the most important components of our Adaptation Manager. While we have not gone into much depth in all parts of the system, we have realized the whole workflow as explained in chapter 3.2. Some subcomponents need more refinement (see section 4), but the implementation shows the suitability and usability (see chapter 3.4) of the Adaptation Manager in general.

The implementation is based on Java J2SE, using an Apache Derby database to store user profiles and device configurations. The application runs as a Java servlet in Apache Tomcat and produces Web pages that can be accessed by a (mobile) Web browser. The Google client was realized using the Google SOAP Search API (see http://code.google.com/apis/soapsearch/) and JDOM 1.0 to parse the XML Web Service files.

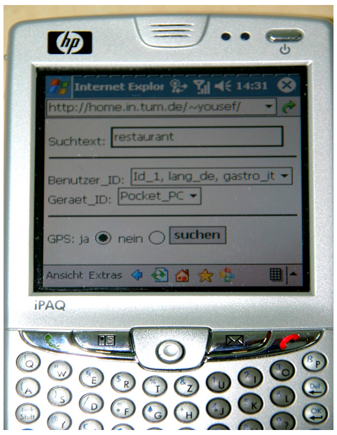
3.4 Test

Figures 3 and 4 show the user interface of our prototype implementation. The test device is a HP iPAQ hw6510 pocket PC running MS Windows Mobile 2003 operating system, even though it ought to work on any device that can display XHTML Mobile 1.0 code.

In this demo, the user can enter a search phrase (Fig. 3, top) and then select among several options. In our demonstration user interface, the user can select from some predefined user ID’s for test purposes. In this example, a user profile will be used that contains information about the user’s favourite type of restaurant (Italian, in this case), which will be added to the search query. Our test user also has the preference to get Web sites in German as search results only, which works fine with the Google API. In a real world application, the search options a user can choose from could be configurable by the user herself as part of her profile.

Another test option is to select the type of device, e.g. “Pocket PC”. This information could be figured out automatically by using a (small) J2ME or Windows Mobile application (instead of the XHTML form), which could also retrieve the GPS coordinates. This is part of the context acquisition that is done on the Client.

Last but not least, the user can specify whether the GPS coordinates are actually used for the query, because searching without location information might produce better results in some cases.



http://home.in.tum.de/~yousef/

Suchtext: restaurant

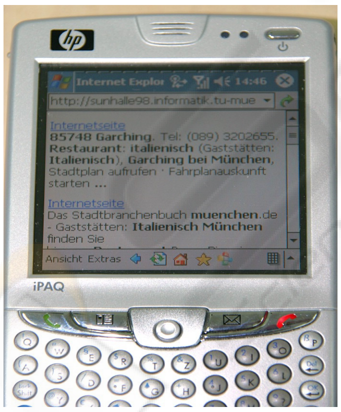
Benutzer_ID: Id_1, lang_de, gastro_it

Geraet_ID: Pocket_PC

GPS: Ja [✓ ●] nein ○ suchen

Figure 3: Search form.

After clicking "suchen", the search will be performed and the results are displayed in XHTML (Fig. 4). A user can then follow the links to the restaurant Web pages, for example. Searching for "restaurant" actually produces links to Italian restaurants in the vicinity of the provided GPS coordinates. This is potentially much better than just trying to manually search since results are location-based and pre-filtered to users' preferences.



http://sunhalle98.informatik.tu-mue

Internetseite
85748 Garching. Tel: (089) 3202655. Restaurant: italienisch (Gaststätten: Italienisch), Garching bei München, Stadtplan aufrufen · Fahrplanauskunft starten ...

Internetseite
Das Stadtbranchenbuch muenchen.de - Gaststätten: Italienisch München finden Sie

Figure 4: Search results.

We have also implemented layout adaptation for standard browsers on PC's. In this case, more search results are shown, more text is displayed per result and also full links are shown. In the above example on the pocket PC, only the term "Internetseite" (German for "Web page") is used to mark-up the link to the actual Web pages, because long URL's could corrupt the layout.

4 EVOLVING RESEARCH ISSUES

We present the following research issues as an extended future prospects section. We are currently working on these issues and are refining our Adaptation Manager to incorporate more "intelligent" functionalities as in our existing prototype implementation, but within the explained architecture and overall design of our approach.

One question is how to better adapt to end user devices. In our approach, every device has a configuration entry in the `ResultEvaluator`. An interesting idea is to check whether search results are suitable for the current end user device or not. This can be done in different ways. First, the `ResultEvaluator` could check the header of the Web pages search results for valid XHTML meta tags. Another possibility is to query mobile link directories. Thereby, the powerful ranking algorithms by Google and other general purpose search engine can be used to generate results, but the results are double checked with specialized mobile directories. This filtering of results could be made optional by allowing the user to configure what kind of Web pages she likes to receive as search results. Another aspect is the dynamical adaptation of the actual Web pages for mobile access, which is investigated as part of the W3C Mobile Web Initiative (see http://www.w3.org/Mobile/), for example, and not part of our project.

In our approach, the rule system which chooses a particular service for a request is rather simple. This could be improved by building a database of services with service descriptions using Semantic Web ontologies (Davies, Fensel and van Harmelen, 2002). For example, if the user searches for a "restaurant", it may be determined that "restaurant" is a sub-concept of "point of interest (POI)" and a POI service could be used to answer the query. Another possibility is a combination of services to further improve search results.

Another interesting research question is the user model and personalization process. In our implementation some user profile attributes are added to the search request based on the keyword(s) the user entered. We are also experimenting with the application of collaborative filtering and a hybrid recommender system (Burke, 2002) to generate personalized information. Thereby, we have realized an application recommender, which recommends Client-side, mobile applications based on the context of the user. A simple example is to recommend a mobile train timetable if the user is situated at a train station, because other users have used this application before in a similar context.

5 SUMMARY AND RELATED WORK

In this paper, we have presented an Adaptation Manager which customizes a user's Web searches, especially for a mobile scenario. We have explained the design of the system. We have implemented the most important components of our design and shown how this approach may be useful to access Google's Web Service. We have also identified several research issues we are working on at the moment.

The main benefits of our approach can be summarized as follows. Firstly, user profiles can be reused for different services since they are stored apart form the services, and users can utilize their preferences for different devices. We also identified personalization components as parts of our Adaptation Manager, which may also be useful in other scenarios. In addition, the implementation can serve as a test bed to further investigate the issues outlined in section 4.

Google offers personalized (see http://www.google.com/psearch) as well as mobile (http://mobile.google.com resp. http://www.google.com/xhtml) search interfaces itself. However, the existing services do not make our approach dispensable but complements our solution. Existing services can be integrated in our Adaptation Manager as explained in this paper. One already mentioned advantage of our approach is that user profiles can be reused for different services, for example other search engines like Yahoo. Managing user profiles apart from the services that are using them may also decrease privacy concerns of users, because users have possibly more control about what information is collected and stored about them. It is also possible to incorporate a privacy manager to control access to the profiles (Woerndl, 2004).

There is related work dealing with the adaptation to people's context, often for specialized scenarios such as context-aware management of meeting rooms. One excellent paper is (Zimmermann, Specht and Lorenz, 2005) and focuses on the problem of context management. Their framework integrates user and context modelling, which is comparable to our solution. The paper introduces a framework and tools for designing context-aware applications. However, their scenario is somewhat different to ours. While Zimmermann et.al. implemented an audio-augmented museum environment as an example, i.e. Client-side, context-aware personalization, we propose a Server-based adaptation of information retrieval.

With regard to query expansion for personalization, (Akrivas et al., 2002) present a technique to context-sensitively expand a user's query using a fuzzy thesaurus. However, their approach is tailored towards the retrieval of documents. Finally, (Keenoy and Levene, 2005) give an overview of research approaches to personalize Web searches and describe their own system "PResTo!". This system is an Internet Explorer plug-in which uses Client-side user profiles to provide a personalized ranking of results from multiple search engines. In a mobile scenario, we think our Server-based approach is more appropriate because of the limitations of mobile end user devices.

REFERENCES

Akrivas, G., Wallace, M., Andreou, G., Stamou, G., Kollias, S., 2002. Context-sensitive semantic query expansion. In: *Proc. International Conference on Artificial Intelligence Systems (ICAIS 2002)*, IEEE.

Brügge, B., Dutoit, A., 2004. *Object oriented software engineering*, Pearson Prentice Hall, Upper Saddle River, USA, 2nd edition.

Burke, R., 2002. Hybrid Recommender Systems: Survey and Experiments. In: *User Modeling and User-Adapted Interaction (UMUAI),* Vol. 12, Nr. 4, pp. 331–370, Springer, Heidelberg.

Davies, J., Fensel, D., van Harmelen, F., 2002. *Towards the Semantic Web. Ontology-driven Knowledge Management*. John Wiley and Sons.

Dey, K.A., Abowd, D., Salber, D., 2001. A Conceptual Framework and a Toolkit for Supporting the Rapid Prototyping of Context-Aware-Applications. In: *Human Computer Interaction*, Volume 16, pp. 97-166.

Keenoy, K., Levene, M, 2005. Personalisation of Web Search. In: Anand, S., Mobasher, B. (eds.): *Intelligent Techniques for Web Personalisation*, Springer LNCS 3169, Heidelberg, pp. 201-228.

Woerndl, W., 2004. Authorization of User Profile Access in Identity Management. In *Proc. IADIS International Conference WWW / Internet 2004*, Madrid, Spain.

Woerndl, W., Groh, G., Toni, K., 2006. Semantic Blogging Agents: Weblogs and Personalization in the Semantic Web. In *Proc. AAAI Spring Symposium on Computational Approaches to Analyzing Weblogs*, Stanford, USA.

Zimmermann, A., Specht, M., Lorenz, A., 2005. Personalization and Context Management. In *User Modeling and User-Adapted Interaction (UMUAI)*, Vol. 15, No. 3-4, pp. 275-302, Springer, Heidelberg.