

# Exploiting Modern Learning Algorithms for Contextual Recommendations

Christian Räck, Stefan Arbanowski  
Fraunhofer FOKUS  
{raeck, arbanowski}@fokus.fraunhofer.de

Stephan Steglich  
TU-Berlin  
steglich@tu-berlin.de

## Abstract

Identifying correlations between context data, user behavior, and semantic information can lead to new services that are able to adapt to different situations. This “personalization” process can be based on recommendations on content. To better support service developers in focusing mainly on the creation of their service logic, these recommendations should be provided by a generic multipurpose recommender. Therefore, this paper proposes a generic framework that delivers “contextual recommendations” that are based on the combination of previously gathered user feedback data (i.e. ratings and clickstream history), context data, and ontology-based content categorization schemes. This paper provides a detailed overview of the specification, a short description of a possible usage scenario, and a discussion of the results.

## 1. Introduction

The World-Wide Web (WWW) provides access to more than 70 million web sites in 2005 [1]. On a wide range of topics the WWW has become the primary source of information. At the same time it has become increasingly difficult to find what we are looking for [2]. Today, search engines are used by most of us to address this issue.

Modern search engines allow us to find all information such as web pages, products, or news that match explicit queries rather easily; queries based on fuzzy keywords lead to large lists of mostly irrelevant search results. If we do not know much about the information we expect to find beforehand, most of us will have difficulties specifying the required queries.

Today’s recommender systems attempt to find web pages, products, or news that are relevant for us. Their recommendations are based on data about our past behavior or statistical models. The resulting data,

which is used to make recommendations, is stored in a user profile. Kobsa [3] and Stewart [4] provide a good survey of user modeling techniques and De Roure [5] provides a review of recommender systems.

The learning algorithms and prediction methods used by today’s recommender systems are strictly tailored to the service for which the system was designed. Moreover, these systems abstract from the prevailing situation during the learning process. This means that they could not provide reasonable recommendations, if the user employs the same service in different situations (such as ‘being at home’ or ‘being at work’) and acts differently.

This paper presents an up-to-date overview of the on-going work on the AMAYA recommender system. At the current stage of our work the focus lies on the analysis and design of the applied approach, i.e. that there are only preliminary results on the quality of the used learning algorithms and prediction methods available.

## 2. Recommender systems

Content-based recommender systems base their recommendations on the similarity between new items and items that a user has liked before (user-to-item). Examples of content-based recommender systems are Fab [6] and ELFI [7]. Fab recommends web pages and

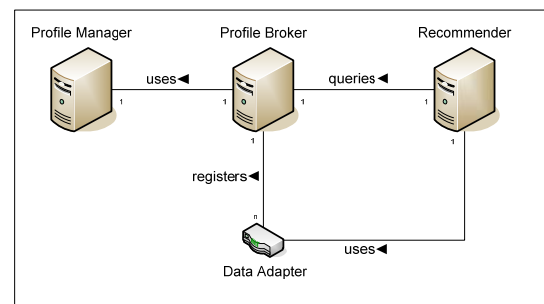


Figure 1. AMAYA recommender system

ELFI recommends funding information from a database.

*Collaborative recommender systems* derive their recommendations on the basis of content ratings from people with similar interests (either user-to-user or item-to-item). PHOAKS [8] and Group Lens [9] are examples of such recommender systems. PHOAKS recommends web links found in newsgroups. Whereas the Group Lens system recommends single newsgroup postings.

*Hybrid recommender systems* tend to combine the advantages of content-based and collaborative recommenders. The feedback required for content-based recommendation is shared, allowing collaborative recommendation as well. Quickstep [10] is an example of a hybrid recommender system. It provides recommendations on research papers based on correlations between user interest profiles and previously classified paper topics. The interest profile is built by processing explicit feedback and a history of browsed topics from each user.

Calvin [11] is a *personal information agent recommender* that monitors the user's browsing behavior over time. It learns to identify broader contexts by relating documents that tend to be accessed together. Calvin extracts information about how the user tends to access groups of documents over time, and predicts document relevance based on similarity of the extended sessions within which various documents are accessed.

### 3. AMAYA recommender system

The AMAYA recommender system is built on two basic requirements: The first requirement is that the system must **provide recommendations for all situations** that might arise in a user's life<sup>1</sup>. And the second requirement is that the system must **support any number of services** a user might want to use.

The first requirement will be met by mapping all personalization data to specific situations in order to be processed by the system. The second requirement will be met by providing a generic interface to a number of learning algorithms and prediction methods as well as introducing an ontology-based content categorization scheme.

AMAYA consists of four functional components: the DATA ADAPTER subsystem, the PROFILE MANAGER subsystem, the PROFILE BROKER subsystem, and the actual RECOMMENDER subsystem.

<sup>1</sup> In regard with the user's interaction with the system and its services.

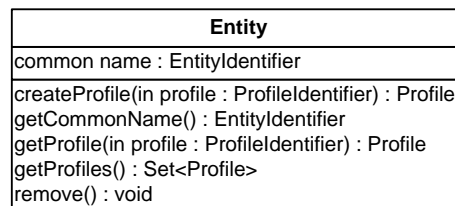


Figure 2. 'Entity' class diagram

A DATA ADAPTER enables the retrieval and processing of distributed personalization data by providing an interface that abstracts from the actual technology used to store the data. The PROFILE MANAGER groups the personalization data in terms of profiles, which provide a mapping to one or more specific situations. The PROFILE BROKER allows querying of all personalization data that is needed in a specific situation. The RECOMMENDER supports the learning of contextual preferences by providing a generic interface to multiple learning algorithms and prediction methods. These algorithms still prevail from the situation during the learning process. However, all learnt models are linked to the current situation during the learning process. The contextual retrieval of these models enables us to provide recommendations for specific situations.

Together these four components provide the required functionality enabling **contextual preferences management** and **contextual recommendations**. Räck [12] provides a detailed overview of the AMAYA approach. Figure 1 shows the basic architecture of the AMAYA recommender system.

### 4. AMAYA data model

The AMAYA data model consists of three major objects: ENTRY, ENTITY, and PROFILE. It shows how any kind of personalization data can be described.

#### 4.1. Entity

ENTITY objects manage all basic information about users or groups.

Each ENTITY is identified by a URI<sup>2</sup>. In addition to that, aliases can be defined in order to enable users to address the record in a more convenient way. This record contains descriptive information (e.g. full name, a description) about that entity and acts as a starting point for the retrieval of all other profile information. Figure 2 shows the corresponding class diagram.

<sup>2</sup> Unique Resource Identifier

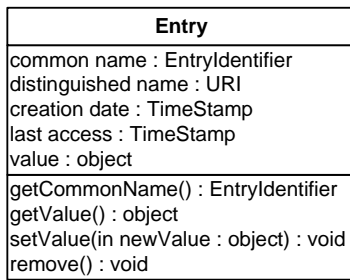


Figure 3. 'Entry' class diagram

## 4.2. Entry

A single unit of personalization data is called ENTRY. An ENTRY object is uniquely identified by its distinguished name [13]. The distinguished name is stored as a URI. In addition to its common name, each ENTRY object stores a value or a list of values. Figure 3 shows the ENTRY class diagram.

The distinction between distinguished names and common names allows the grouping of different ENTRY objects by introducing additional hierarchy levels. Such a group is called component according to the CC/PP recommendation [14] and the GUP specification [15]. The key difference between this data model and the previously mentioned approaches lies in the fact that AMAYA provides contextual profiles (i.e. here: more than one profile) whereas the other approaches provide only one profile per user (or per device).

A set of ENTRY objects will be provided by a single DATA ADAPTER and consists of multiple generic entity preferences as well as a set of service-specific entity preferences.

## 4.3. Profile

A PROFILE groups multiple units of personalization data and links them to a specific situation that describes a user or a group in a **precise interaction context**. Several PROFILE objects can be attached to an ENTITY.

Usually, a PROFILE describes just one subset of all possible situations in which a user or a group can be encountered. There are situations like *'Bob being at home'*, *'Bob using certain terminals'*, *'Bob accessing information at a certain time'*. A PROFILE may also contain any reasonable combination of the situations described above. Each PROFILE is uniquely identified by a distinguished name. Figure 4 shows the PROFILE class diagram.

Mapping of ENTRY objects to specific situations is done by a set of qualifiers, which specify all conditions

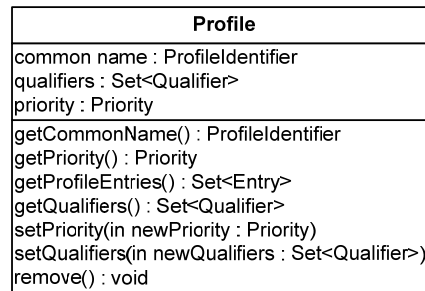


Figure 4. 'Profile' class diagram

that have to be met to consider a PROFILE valid for a specific situation. Besides these qualifiers, all profiles have to be ordered so that the PROFILE BROKER is able to identify a single PROFILE (or none at all), which is returned as the result of a query in a specific interaction context.

A PROFILE may be enhanced with additional functionality, which can be plugged-in separately, e.g. Middleton [16] considered simple time-decay functions as an easy to use profiling technique that can be applied to content and rating-based profile representations. A PROFILE object could take care of the automatic aging of ENTRY objects in this way.

## 5. AMAYA component structure

AMAYA consists of four functional components: the DATA ADAPTER subsystem, the PROFILE MANAGER subsystem, the PROFILE BROKER subsystem, and the actual RECOMMENDER subsystem.

### 5.1. Component: DATA ADAPTER

A DATA ADAPTER wraps several personalization data units by providing a uniform data access interface for retrieval and storage of this information. Each DATA ADAPTER registers at the PROFILE BROKER once its data sources are ready to provide the data. These data sources can be accessed, e.g. via SQL / JDBC / ODBC in case of a full-fledged database or via REST<sup>3</sup> [17] in case of an embedded database running on a device with limited resources such as a mobile phone or PDA. This means that, if the personal preferences are stored on a mobile device, they are also available when the device is not connected to a network.

### 5.2. Component: PROFILE MANAGER

The PROFILE MANAGER governs all entities that require contextual personalization data. All internal

<sup>3</sup> Representational State Transfer

data regarding users and groups including the mapping of their preferences to specific situations is stored here. For each user or group there is a record called ENTITY and a set of PROFILE objects.

The PROFILE MANAGER provides an interface that can be used by a service or by the PROFILE BROKER in order to process an ENTITY and all PROFILE objects that belong to the record. The personalization data itself is accessed via a DATA ADAPTER, which is returned by the PROFILE BROKER along with a list of ENTRY ids.

It is also possible for an entity to decide to bypass the PROFILE BROKER's decision making process by manually specifying a PROFILE, which should be used for all queries that may follow. In addition to that, each PROFILE can also be excluded from future queries by manually disabling it.

### 5.3. Component: PROFILE BROKER

The PROFILE BROKER retrieves and processes all queries for contextual personalization data. In order to process these queries the PROFILE BROKER has to check whether there is a proper match (i.e. PROFILE) to this query or not (i.e. that the resulting PROFILE has to match the situation description that has been specified in the query). If a PROFILE is found, the PROFILE BROKER returns a set of ENTRY objects together with the required DATA ADAPTER objects to retrieve the

data. The next sections describe the matchmaking algorithm that has been implemented.

A service that wants to personalize itself specifies the present situation of a user and requests from the PROFILE BROKER all preferences, which are relevant. The PROFILE BROKER uses the PROFILE MANAGER to get a list of PROFILE objects, which belong to that entity, and tries to find a matching PROFILE given the description of the entity's situation. If a PROFILE is found, the broker returns a set of ENTRY objects, which are identified by the PROFILE, together with their DATA ADAPTER objects. The service has to use the provided DATA ADAPTER objects to retrieve the preferences values directly. It has to use the DATA ADAPTER objects to update the previously received preferences values, too.

A set of privacy enforcement rules can be applied by a PEP<sup>4</sup> so that a service gets only preferences that it is allowed to receive (as stated by the user and his / her definition of appropriate policies).

Figure 5 provides an overview of the preferences retrieval process. Please note that all DATA ADAPTER objects have been removed from the sequence diagram for a better readability.

The following matchmaking algorithm is used by the PROFILE BROKER to process all queries: Each PROFILE is given a priority by the PROFILE MANAGER. This priority is used to sort all profiles from the highest priority to the lowest priority. This search order is used

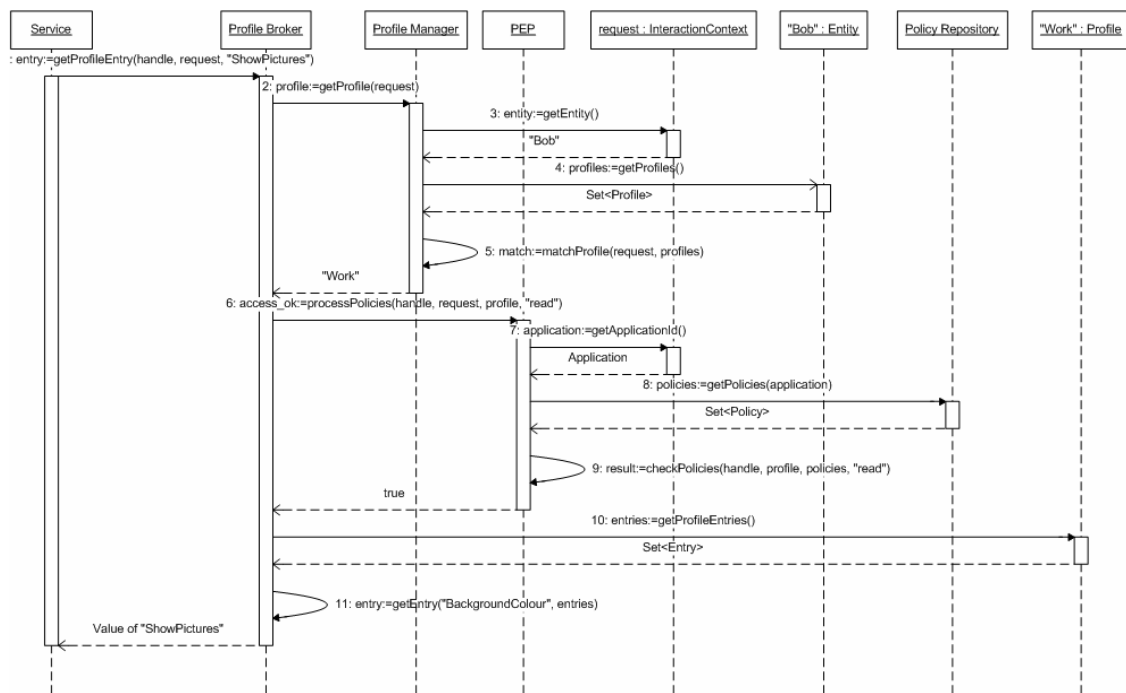


Figure 5. 'Retrieval of user preferences' sequence diagram

to determine the most appropriate PROFILE for a given interaction context. This means that the first PROFILE that matches the provided description is returned. The PROFILE qualifiers are used to determine whether a specific PROFILE matches the description or not. The sort sequence guarantees that the most relevant PROFILE is found, if any. This search algorithm also ensures that all conflicts (i.e. several profiles match the provided interaction context) will be solved.

There are two ways of providing the required description of an entity's situation. On the one hand the service can provide a simple description of the situation itself. On the other hand the description can be provided by an external component that manages all context data [18], which characterizes the present situation. This frees the service from the need to gather and to infer all context data itself. In the context of a seamless and context-aware computing environment, the second possibility has to be preferred. It enables a more complex characterization of the situation.

The RECOMMENDER employs the PROFILE BROKER to request a set of service-specific data that is updated or processed by learning algorithms and predictions methods, too.

#### 5.4. Component: RECOMMENDER

The RECOMMENDER enables the learning of user and group interests (e.g. 'Bob likes sports news.') and the prediction of their behavior (e.g. 'Since Alice likes

Chinese food and since her interests are very similar to Bob's interests, it's very likely that Bob likes Chinese food, too.').

All recommendations are provided on the basis of a **two-step process**: At first a *user model has to be learnt*. On this basis a *prediction / recommendation* can be made. Figure 6 shows the learning of user preferences.

Content-based filtering algorithms and collaborative filtering algorithms allow the learning of service-specific interests based on either implicitly or explicitly gathered feedback.

Explicitly gathered feedback is provided by means of positive or negative ratings, which is directly received from an entity (i.e. every time Bob rates a recipe received by a cookbook service, this rating is sent to the RECOMMENDER by the service in behalf of Bob). The service appends a description of the entity's present situation to the received feedback (e.g. 'Bob is driving in his car.'). so that the RECOMMENDER can request the appropriate entity preferences before updating them. The request is carried out with the help of the PROFILE BROKER.

AMAYA requires implicitly gathered feedback (e.g. clickstreams) to be expressed in terms of a positive or negative rating by the service (instead of the user).

In order to provide a service-independent interface to a number of learning algorithms and prediction methods the RECOMMENDER introduces the concept of a CONTENT ITEM, which describes an atomic unit of

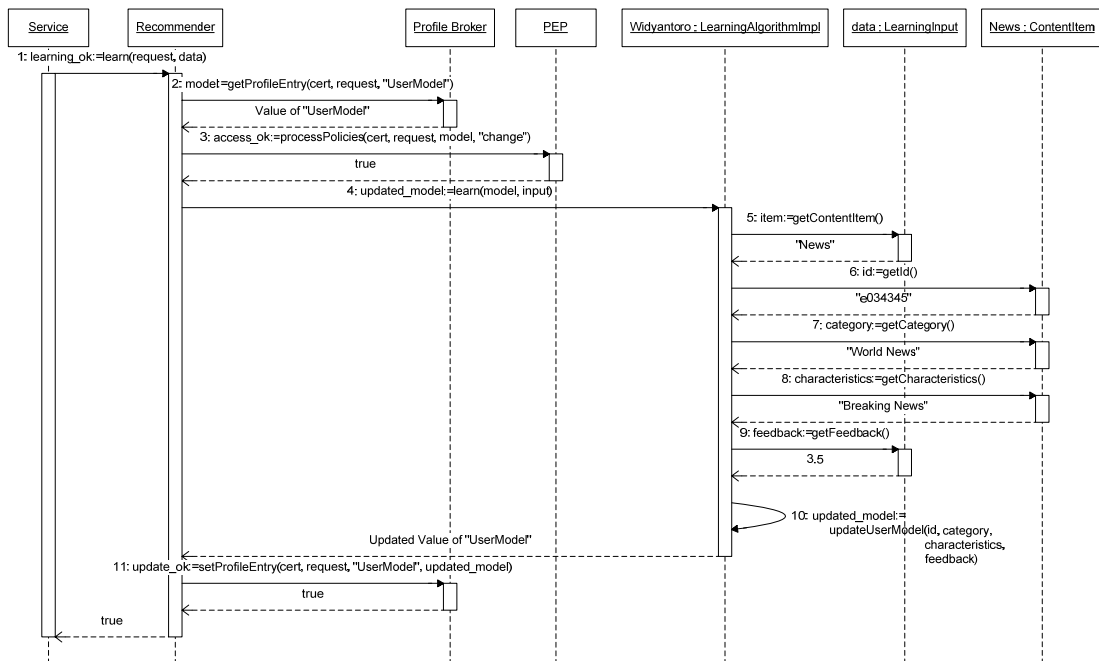


Figure 6. 'Learning of user preferences' sequence diagram

information (i.e. content) that belongs to a service.

A CONTENT ITEM abstracts from the actual information (e.g. ‘Google, NASA Partner on Research’ or Alicia Keys’ CD ‘Songs in A minor’) and provides a description that can be used for learning purposes. This description consists of a uniform resource identifier and a set of distinct features such as service-specific categories and other content-related characteristics (e.g. further key words).

The service-specific categories are based on an ontology-based content categorization scheme. This scheme allows expressing different content categories (e.g. ‘sport news’ and ‘business news’) as well as all relations between different content categories (e.g. ‘soccer news is a sub-category of sport news’).

Basically, the interface of the RECOMMENDER provides two methods that can be used by services:

The first method is used to provide input on a per service basis. The input is used during the learning process. It is received as either positive or negative feedback on specific CONTENT ITEM objects.

The second method is used to answer the question whether a specific CONTENT ITEM is relevant to an entity in a specific situation or not.

All learning algorithms and prediction methods have to implement the same abstract interface. With the help of this interface all learning algorithms and prediction methods provide their functionality transparently to a service. A service developer doesn’t have to care about any algorithm-specific and implementation-specific details. All what is required is to choose the algorithm that suits a service best.

## 6. Usage scenario

A car rental agency or a hotel chain wants to provide a new value-added service to their high profile customers. The underlying assumption for this usage scenario is that this customer group appreciates a high level of convenience and that they are willing to pay an extra fee to reach that level.

The basic idea is that the product offered by the car rental agency (*cars*) or the hotel chain (*business suites*) can be adapted to the present situation of a customer. Preferences may include, among other things, preferred environment settings such as preferred temperature ranges or individual tastes in music.

## 7. Discussion

The AMAYA RECOMMENDER subsystem does not care about the mapping of preferences to specific situations.

An advantage of this approach is that well-known learning algorithms and prediction methods can be used without any modifications. Rocchio’s relevance feedback algorithm [19] can now be used to infer the relevance of information according to an entity’s situation.

A drawback of this approach is that all situations that are available for learning purposes have to explicitly be defined by PROFILE qualifiers. Complex coherences between preferences and situations, which are normally not directly perceived by an entity, are learnt with a great effort only or not at all (i.e. as long as Bob is not aware<sup>5</sup> of the fact that he acts differently when ‘being at home’ or ‘being at his office’, the RECOMMENDER cannot learn the differences between both situations).

## 8. Evaluation and Conclusion

A first trial of AMAYA and a web-based news service showed good results so far. The implemented learning algorithm utilized a three-descriptor vector to learn changes in user interests over time [20].

Different user interests in predefined situations such as ‘being at work’ or ‘being at home’ had been learnt successfully. Based on the recommendations provided by AMAYA our news service delivered convenient information for each user.

## 9. Future work

The next steps for this work are to run more trials based on several popular service scenarios and to implement different learning algorithms and prediction methods.

The future evolution path of our contextual recommender system approach leads to the question on how the quality of the provided personalization data can be improved in order to allow better personalization results. To answer this question it has to be evaluated how additional information / more detailed user models can be inferred from the readily available data.

The idea of using the readily available entity preferences to infer additional personalization data could improve service scenarios where new or unused services suffer from the cold-start problem [21]. This could allow delivering of a better overall quality of the personalized service to the user. The usage of further

---

<sup>5</sup> Bob can express his awareness by specifying different PROFILE objects for both situations.

ontological models as a basis for this process has to be analyzed.

Another research direction leads to the evaluation of knowledge-based approaches to user modeling. The usage of current and past data to engineer static entity models could be applicable for the learning of contextual entity preferences, too. This allows the inference of new situations (PROFILE qualifiers), i.e. that new PROFILE objects and appropriate qualifiers could be inferred by these approaches.

## References

- [1] Hobbes' Internet Timeline [Online]. Available: <http://www.zakon.org/robert/internet/timeline/>
- [2] D. Shenk, "Data Smog: Surviving the Information Glut", *Technology Review*, May/June 1997.
- [3] A. Kobsa, "Generic User Modeling System", *User Modeling and User-Adapted Interaction*, issue 11, pp. 49 – 63, Kluwer Academic Publishers, 2001.
- [4] A. Stewart, C. Niederée and B. Mehta, "State of the Art in User Modelling for Personalization in Content, Service and Interaction", *NSF/DELOS Report on Personalization*, 2004.
- [5] D. De Roure, W. Hall, S. Reich, G. Hill, A. Pikrakis, M. Stairmand, "MEMOIR – an open framework for enhanced navigation of distributed information", *Information Processing and Management*, issue 37, pp. 53 – 74, 2001.
- [6] M. Balabanovi, Y. Shoham, "Fab: Content-based, collaborative recommendation", *Communications of the ACM*, volume 40, no. 3, 1997.
- [7] I. Schwab, W. Pohl, I. Koychev, "Learning to Recommend from Positive Evidence", *Proceedings of Intelligent User Interfaces*, pp. 241 – 247, ACM Press, 2000.
- [8] L. Terveen, W. Hill, B. Amento, D. McDonald, J. Creter, "PHOAKS: a system for sharing recommendations", *Communications of the ACM*, volume 40, no. 3, 1997.
- [9] J. Konstan, B. Miller, D. Maltz, J. Herlocker, L. Gordon, J. Riedl, "GroupLens: applying collaborative filtering to Usenet news", *Communications of the ACM*, volume 40, no. 3, 1997.
- [10] S. Middleton, D. De Roure, and N. Shadbolt, "Capturing knowledge of user preferences: ontologies in recommender systems", *Proceedings of the First International Conference on Knowledge Capture (K-CAP)*, Victoria, B.C., Canada, Oct. 2001.
- [11] T. Bauer, D. Leake, "Exploiting information access patterns for context-based retrieval", *Proceedings of the 7th international conference on Intelligent user interfaces*, San Francisco, California, USA, pages 176-177, ACM Press, 2002.
- [12] C. Räck, S. Steglich, S. Arbanowski, "AMAYA: A Recommender System for Ambient-Aware Recommendations", *Proceedings of the 2005 International Conference on Internet Computing (ICOMP)*, Las Vegas, Nevada, USA, June 27 – 30, 2005.
- [13] ITU Telecommunication Standardization Sector (ITU-T) X.500, "The Directory: Overview of concepts, models and services", ISO/IEC 9594-1.
- [14] World Wide Web Consortium (W3C), "Composite Capability / Preference Profiles (CC/PP): Structure and Vocabularies", W3C Recommendation, January 2004.
- [15] 3rd Generation Partnership Project (3GPP) TS 23.240, "3GPP Generic User Profile (GUP) Architecture", version 6.6.0, December 2004.
- [16] S. Middleton's Dissertation: [Online], chapter 3, available: <http://www.ecs.soton.ac.uk/~sem99r/>
- [17] R. Fielding's Dissertation: [Online], chapter 5, available: [http://www.ics.uci.edu/~fielding/pubs/dissertation/rest\\_arch\\_style.htm](http://www.ics.uci.edu/~fielding/pubs/dissertation/rest_arch_style.htm)
- [18] S. Steglich, C. Räck, and S. Arbanowski: "Ambient-Aware Information Delivery for Beyond 3G Systems", *4th Workshop on Applications and Services in Wireless Networks (ASWN)*, Boston, Massachusetts, USA, August 9 - 11, 2004.
- [19] J. Rocchio, "Relevance Feedback in Information Retrieval", *The SMART Retrieval System: Experiments in Automatic Document Processing*, pp. 313 - 323. Prentice-Hall, Englewood Cliffs, NJ, 1971.
- [20] D. Widyantoro, T. Ioerger, J. Yen, "Learning User Interest Dynamics with a Three-Descriptor Representation", *Journal of the American Society for Information Science (JASIS)*, 2000.
- [21] J. Herlocker, J. Konstan, L. Terveen and J. Riedl, "Evaluating collaborative filtering recommender systems", *ACM Transactions in Information Systems*, volume 22, issue 1, pages 5-53, ACM Press, 2004.