Artificial Intelligence Review **18:** 33–74, 2002.

User Modeling for Personalized City Tours[1]

JOSEF FINK[1] and ALFRED KOBSA[2]
[1]*humanIT–Human Information Technologies, Sankt Augustin, Germany (E-mail: Josef.Fink@humanIT.com);* [2]*Dept. of Information and Computer Science, University of California, Irvine, USA (E-mail: kobsa@uci.edu)*

Abstract. Several current support systems for travel and tourism are aimed at providing information in a personalized manner, taking users' interests and preferences into account. In this vein, personalized systems observe users' behavior and, based thereon, make generalizations and predictions about them. This article describes a user modeling server that offers services to personalized systems with regard to the analysis of user actions, the representation of assumptions about the user, and the inference of additional assumptions based on domain knowledge and characteristics of similar users. The system is open and compliant with major standards, allowing it to be easily accessed by clients that need personalization services.

Keywords: interest profile, LDAP, learning about the user, mobile tourist guide, personalization, user modeling server

1. Introduction

Computer support for travel and tourism has recently attracted considerable interest, both with regard to research and experimental deployment. Possible assistance for travelers and tourists ranges from web-based travel planning to stationary information kiosks and then on to location-aware portable museum and city guides. Since tourism is intimately connected with personal interests and preferences, many of the systems developed in this area aim at providing information in a *personalized* manner (Abowd et al. 1997; Fink et al. 1998; ILEX 1998; Not et al. 1998; Cheverst et al. 2000a; Malaka and Zipf 2000; Oppermann and Specht 2000; Poslad et al. 2001). Personalization means that systems cater to each individual user, thereby taking e.g. his interests, preferences and background knowledge into account. As a prerequisite, personalized systems must be able to watch the user's behavior and make generalizations and predictions about the user based on their observations. This information about him is usually collected in a so-called *user model* and administered by a *user modeling system* (Wahlster and Kobsa 1989; Kobsa et al. 2001).

The work presented here was carried out in the context of the Deep Map project (Malaka and Zipf 2000; Deep Map 2001) of the European Media

Laboratory in Heidelberg, Germany. It also benefited from prior experience in the AVANTI project (Fink et al. 1998). Deep Map is part of a family of long-term research projects aimed at developing personal web-based and mobile tourist guides, thereby integrating research from various areas of computer science: geoinformation systems, data bases, speech input and output, multilinguality, intelligent user interfaces, knowledge representation, and user modeling (see (EML 1999; Malaka 1999; EML 2000; Malaka and Zipf 2000) for more details about the project aims). In order to enable Deep Map and other components to provide personalized behavior, a model of relevant characteristics of individual users (namely users' individual interests and preferences) and of user groups is acquired and maintained by the user modeling system of Deep Map.

A central aim of the WebGuide sub-project (WebGuide 2001) is the provision of personalized tour recommendations for the city of Heidelberg that cater to an individual user's interests and preferences. WebGuide identifies geographical points of interest and computes a tour that connects these points via presumably interesting routes based on the following pieces of information:

- geographical information about Heidelberg,
- information about points of interest (e.g. the Heidelberg Castle),
- information about selected means of transport (e.g. car or bike),
- individual users' interests and preferences, and
- tour restrictions specified by the user (e.g. regarding distance and duration).

Tour recommendations that meet the above requirements are then presented to the user. Figure 1 depicts two proposals for walking tours that were prepared for the same user by a first prototype of WebGuide. The tour proposal on the left side does not take individual user interests and preferences into account, while the proposal on the right respects particularly the user's dislike of environmental burden. While both tours contain the same points of interest (indicated by black dots), the proposed routes between these points (depicted by bold lines) partially differ in some areas. In the personalized tour, routes that are presumably problematic with respect to environmental burden (e.g. routes along streets with high traffic) are substituted by more appropriate paths if possible (e.g. by routes through pedestrian zones, parks and forests).

Figure 2 shows prototypes of mobile applications of Deep Map that use Compaq I-PAQs and Xybernaut's Mobile Assistant IV (Xybernaut 2001). These applications are location-aware and can take the user's position into account when catering information to him.

The remainder of this paper is structured as follows: In Section 2, we discuss the requirements for user modeling that we uncovered in this

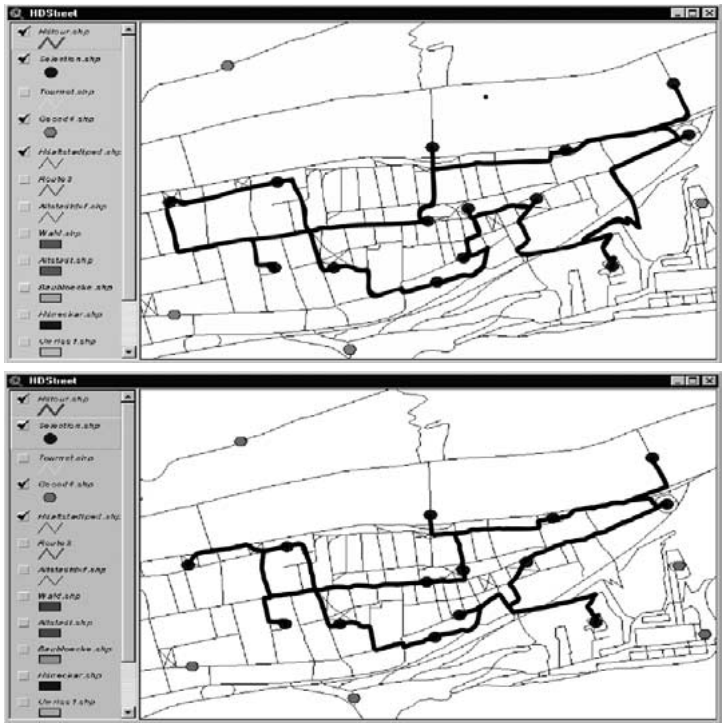

Figure 1. A non-personalized and a personalized tour proposal from WebGuide.[2]

application domain, and give an overview of the generic user modeling system that we developed in order to meet these requirements. Section 3 describes its central Directory Component, which represents and communicates assumptions about the user. Section 4 presents its current User Modeling Components, which draw assumptions about the user based on his information requests, the interaction behavior of other users, and knowledge about the application domain. Section 5 describes administrative tools for accessing the directory component, and Section 6 mechanisms to ascertain privacy and security. Sections 7 and 8 describe related mobile applications and summarize the contributions of this paper.

2. Requirements for User Modeling Systems in a Travel Domain

2.1. *Requirements analysis*

In order to provide user-related information for adaptation purposes, a user modeling system has to carry out the following central tasks for several users at the same time (cf. Kobsa 2001a):

Figure 2. Three mobile applications of Deep Map.[3]

- learn the interests and preferences of users based on their usage of the application,
- predict interests and preferences of individual users based on those of similar users, and on assumptions about homogeneous user subgroups (so-called "stereotypes"),
- infer additional interests and preferences using domain knowledge,
- store, update and delete explicitly provided information and implicitly acquired assumptions,
- care for the consistency and privacy of the user model contents, and
- supply authorized applications with current information about the user.

In addition to these relatively generic requirements, we also elicited the following user modeling requirements in a number of scenario-based design sessions (Carroll 1995, 2000):

- The user characteristics that need to be taken into account include interests and preferences, and selected demographic data (e.g. users' age, gender, and continent of origin).[4]

- A priori knowledge about users (e.g. from other user-adaptive systems or smartcards that store users' interests (Fink et al. 1997)) is generally not available.
- The explicit acquisition of user information at runtime must be restricted to a brief initial interview. The emphasis must lie on the *implicit* (i.e. unobtrusive) acquisition of user interests and preferences, e.g. from user feedback, usage data, models of similar users, and inferences based e.g. on domain heuristics.
- Adaptation should be relatively quick. For instance, the provision of personalized information and services should already be possible during the user's first session with WebGuide. In subsequent sessions, the system should be able to cater to a user's changing interests and preferences.
- Long-term user modeling should be supported (which implies that the lifetime of individual user models in WebGuide must extend beyond a single user session).
- Security and privacy, and related technical implications (e.g. scrutability of user model contents by Deep Map users), should be taken into account.

Other identified requirements result from related user modeling research, or simply conform to best practice in the design of software systems:

- Easy access to the user modeling system should be possible from different applications, and different software and hardware platforms.
- Different user model acquisition techniques should complement each other, and synergistic effects between methods should be exploited. For instance, an acquisition method that predicts user characteristics based on similarities between user profiles should be able to take interests and preferences into account that were explicitly provided by users, as well as interests and preferences that were implicitly acquired by other acquisition methods.
- The user modeling system must be open with respect to
  - new or evolving user modeling requirements and their implications for the acquisition and representation of user models, for learning methods, and for inferences on the basis of user models,
  - external data sources about users that may become available in the future (e.g. P3P profiles (Reagle and Cranor 1999)), and
  - tools for user model analysis (e.g. visualization tools, data mining tools) and associated interface standards like ODBC.
- User models must be upwards compatible between existing and future Deep Map prototypes.

- The quality of service of the user modeling system is very important (e.g. its responsiveness, robustness, and the access management facilities it provides).
- The management of arbitrary information about Deep Map components (their configuration, location, etc.) should be possible within the user modeling system. This enables especially mobile and location-aware applications of Deep Map to flexibly adapt their services to the available computing resources (locally available resources on mobile devices are usually much more limited than server resources through a network).
- The user modeling system should comply as much as possible with existing and emerging standards and *de facto* standards (e.g. from organizations like ISO and IETF).

All these requirements substantially influence the architecture and the properties of a user modeling system. These demands are however not unusual, but fairly typical for personalized information systems that are being used in short interactions only. In the remainder of this paper we will discuss our design of a user modeling system that meets these requirements.

2.2. *Generic user modeling systems*

User modeling systems can be constructed in such a way that they are tightly intertwined with the application system. They are then part of the application, and their functionality is exclusively geared to the demands of the application. Many current personalized systems are designed in this manner (see the overviews in Carberry 2000; Kobsa 2001b). For the purposes of Deep Map, a user modeling system was instead developed that is independent of the specific user-adaptive application with respect to its architecture and the user model contents, but can be easily configured to meet the special needs of such applications. This user modeling system falls under the category of *generic user modeling systems* and, due to its operation as a server, more specifically under the category of *user modeling servers* (Kobsa 2001a). We will from now use the acronym "UMS" to refer to our user modeling system/server, and "UMS for Deep Map" to refer to its special configuration for the purposes of Deep Map.

Existing academic and commercially available user modeling servers were found to lack with respect to (i) acquisition methods, particularly the integration of domain knowledge and the mix of methods at deployment time, (ii) extensibility, and (iii) the integration of user-related information that is external to the user modeling server (Fink and Kobsa 2000; Kobsa 2001a). They were therefore not considered as suitable for Deep Map. The user modeling server that we developed in Deep Map builds on experience from both previous AI-oriented research prototypes and recent commercial devel-

opments (cf. Fink 2002, Chapters 2 and 3). Unlike all former user modeling servers, the UMS does not store user models in knowledge representation systems or database management systems, but rather in *directory management systems* (or *directories* for short). Directories employed by the UMS are based on the LDAP standard (Howes et al. 1999), which in turn is based on the X.500 standard (ITU-T 1993; Chadwick 1996). In contrast to special-purpose directories like network operating system directories (e.g. Novell 2001; Microsoft 2001) or centralized Internet directories (e.g. Bigfoot 2001), LDAP is not limited to a particular purpose, application or operating system. LDAP contains a large number of predefined user-related information types and allows for arbitrary extensions. LDAP directories can be distributed across a network of several servers. The maintenance of multiple and partial copies at several sites and their automatic replication is possible as well, which is a major advantage for mobile applications when the network connection is not fully reliable.

Figure 3 depicts the architecture of the User Modeling System for Deep Map, which consists of the following two components:

- The *Directory Component* comprises three subsystems, which care for the representation of information about the user, for its communication with the User Modeling Components and the External Clients, and the mediation of its interaction with the User Modeling Components. The Directory Component will be described in more detail in Section 3.
- The *User Modeling Components* perform dedicated user modeling tasks. In Deep Map, three clients of the Directory Component are employed: the User Learning Component (ULC) and the Mentor Learning Component (MLC), each of which learns about the user in a different manner; and the Domain Inference Component (DIC), which draws inferences about the user based on domain knowledge. The User Modeling Components will be discussed in Section 4.

The left side of Figure 3 shows External Clients that are "consumers" and "producers" of information contained in the UMS, or access the UMS for administrative purposes. They will be discussed in Section 5. The overall architecture in Figure 3 is generic, while the specific selection of User Modeling Components and External Clients as well as the contents of the representation are specific for Deep Map. The UMS does not mandate any specific distribution topography, but components of the User Modeling System can be flexibly distributed across a network of computers depending on the available computing resources (in Deep Map, however, the Directory Component and the User Modeling Components normally reside on the same computer).

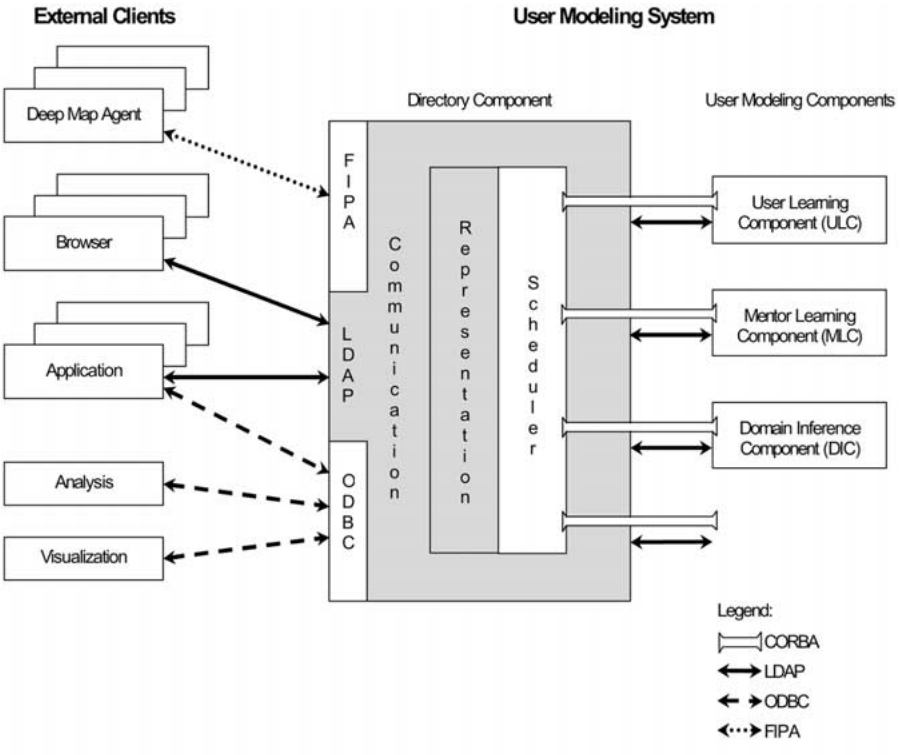

Figure 3. Architecture of the UMS in Deep Map.

3. The Directory Component

3.1. *Representation*

The task of the Representation subsystem of the Directory Component is to store various models. The most important ones will be described in this section. All models hosted by the UMS are based on standard LDAP object class and attribute definitions. The implementation of the Directory Component is based on the Directory Server from (iPlanet 2001).

3.1.1. *User models*

The current version of the UMS for Deep Map hosts a User Model, a Usage Model, a System Model and a Service Model. Figure 4 depicts three user models in the left frame, one for *Peter Smith*, one for *George Brown*, and one for a stereotype called *Art Lover*.[5] In general, user models comprise a demographic part, which is mainly based on standard LDAP object class and attribute definitions, and a part for users' interests and preferences. The right frame of Figure 4 shows the demographic attributes for *Peter Smith* (his entry

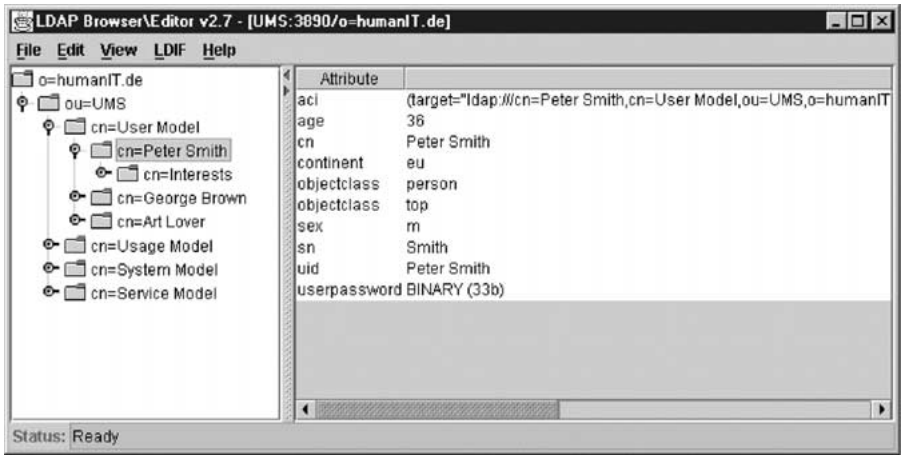

Figure 4. User models.

is selected in the left frame). Inherited attributes from the object class person are the common name and the surname (abbreviated *cn* and *sn*) and optional attributes (e.g. an encrypted *userpassword*). Several other inherited attributes (e.g. description, telephone number) have not been filled with values for Peter Smith yet. Other optional attributes (*age, continent, sex*) have been added to meet the specific needs of WebGuide.

The major portion of a user model is devoted to users' interests and preferences. The topography and terminology of the interest model corresponds to the domain taxonomy of Deep Map, which is maintained in the system model (see Section 3.1.3). Figure 5 depicts the user model of Peter Smith with his *Interests* being unfolded in the left frame. Interests can be hierarchically ordered (e.g. interest in *Nature* comprises interest in *Climate, Flora, Fauna, Scenery*, and *Environmental Burden*).

The interest in *Environmental Burden* is currently selected. User attributes and operational[6] attributes for this entry are shown in the right frame. The most important ones are the following:

- *classification* indicates whether a user's *normalized_probability* (see below) of interest is significantly high, significantly low, or not significant. *Peter Smith*'s interest in *Environmental Burden* was considered to be significantly high (as indicated by the keyword "yes"). This assumption was calculated by the ULC (see Section 4.1 on the ULC).
- *individual_probability* is an assumption about a user's interest based on the types of information she retrieves from the system (see Section 4.1 on the ULC). The probability of *Peter Smith*'s interest in *Environmental Burden* is considered to be quite low (namely 0.04).

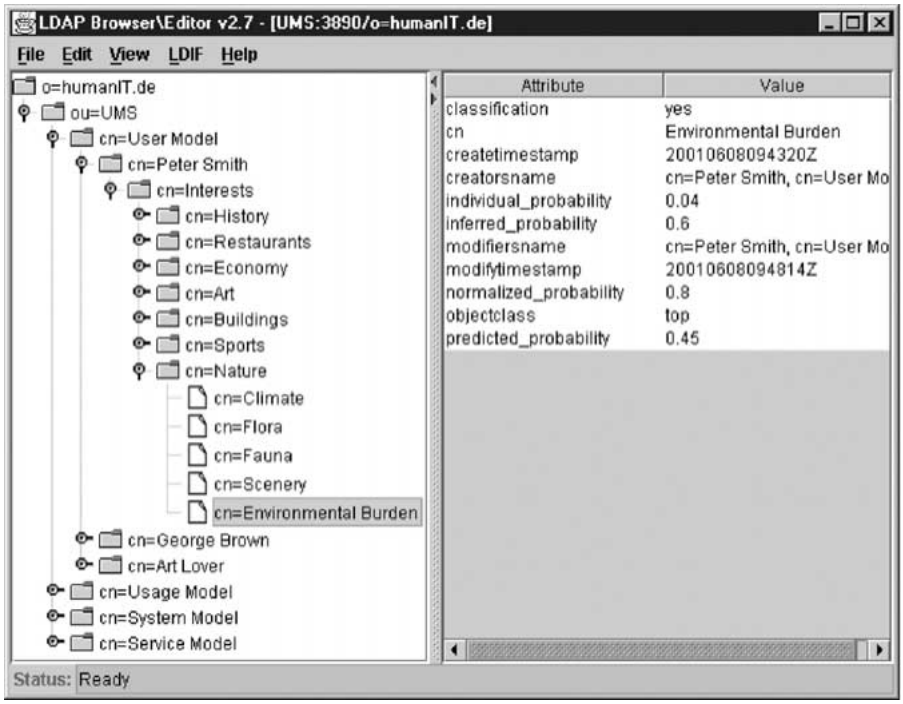

Figure 5. Interest model of Peter Smith.

- *inferred_probability* is an assumption about a user's interest that was acquired by applying domain inferences (see Section 4.3 on the DIC). The inferred probability of *Peter Smith*'s interest in *Environmental Burden* is considered medium (namely 0.6).
- *normalized_probability* rates an individual user's interest, as indicated by *individual_probability*, in relation to the interests of the whole user population. For *Peter Smith*, this probability is considered to be fairly high (namely 0.8) since most other users are presumed to have a much lower interest in *Environmental Burden* than him. This assumption was acquired by the ULC.
- *predicted_probability* is a prediction about a user's interest based on his similarity with other users or stereotypes. In our scenario, *Peter Smith* was found to be similar to the stereotype *Art Lover* (see Section 4.2 on the MLC). This prediction was calculated based on the presumable interest of this user group in *Environmental Burden*, and the degree of similarity between him and this group.
- *creatorsname* indicates the creator of this entry (i.e. the user or a user modeling component).
- *createtimestamp* indicates the time of creation.

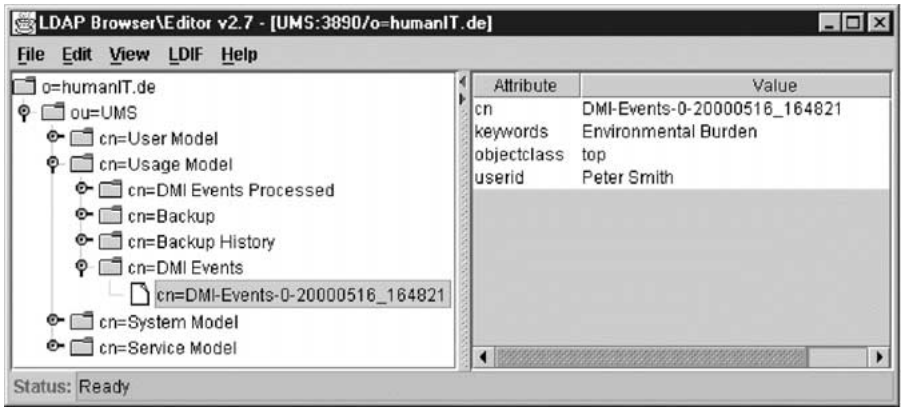

Figure 6. Usage model.

- *modifiersname* indicates the user who last modified this entry (namely Peter Smith himself).
- *modifytimestamp* contains the time at which the last modification occurred.

For more information on the role of the User Model in the user modeling process we refer to Section 4 that deals with learning about the user.

3.1.2. *Usage models*

The Usage Model acts as a persistent storage for usage-related data within the UMS. It comprises usage data communicated by WebGuide, and information related to the processing of these data in user modeling components (e.g. a counter for *Peter Smith*'s interface events related to *Environmental Burden*). In the left frame of the editor depicted in Figure 6, we see the Usage Model from an administrator's point of view. It comprises the following parts:

- *DMI*[7] *Events Processed* includes information that is required for, and results from, processing usage data contained in *DMI Events* (e.g. the aforementioned event counter for *Environmental Burden*).
- *Backup* and *Backup History* may contain events from *DMI Events* that have already been processed by user modeling components. The main motivation for stockpiling interface events is to preserve them for further processing and analysis, e.g. by employing external visualization and data mining tools (see Figure 3).
- *DMI Events* contains usage data communicated by WebGuide. Each entry in this sub-tree describes a WebGuide interface event in terms of one or more interests from the domain taxonomy that can be attributed to the user based on this event. For instance, *Peter Smith*'s request for a document about the environmental impacts of tourism is described

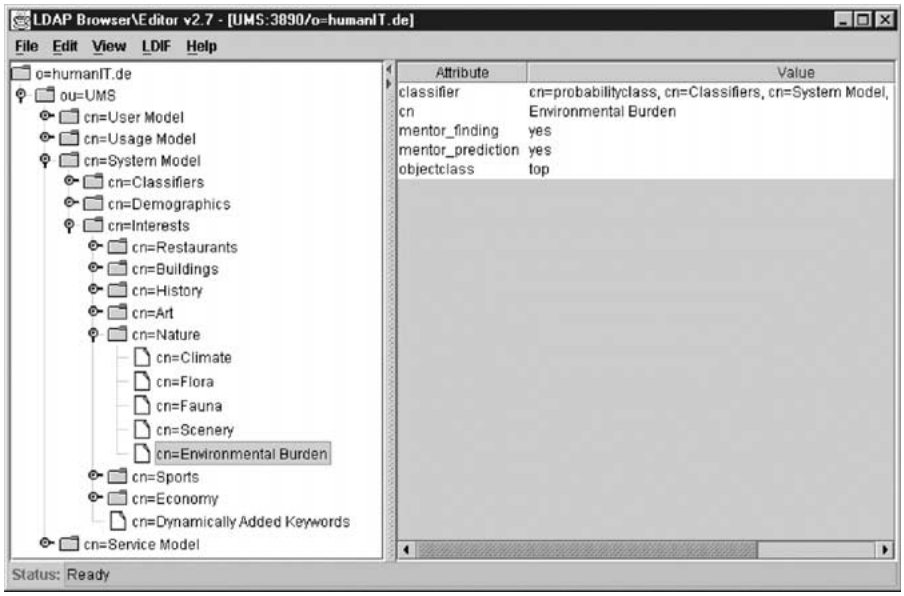

Figure 7. Fragment of the domain taxonomy in the system model.

in terms of an attributed interest called *Environmental Burden*. The currently selected entry in the left frame and the associated information in the right frame illustrate this.

For more information on the Usage Model and its role in the user modeling process we refer to Section 4.1.2 that deals with learning about the user.

3.1.3. *System model*

The System Model encompasses information about the application domain that is relevant for all user modeling components of the UMS. Its most important content is the mentioned domain taxonomy. In the current version of the UMS for Deep Map, the System Model comprises the following parts (also see Figure 7):

- *Classifiers* contains templates that control the discretization of continuous attribute values (e.g. for discretizing users' age into appropriate age groups). The MLC uses these templates for preprocessing attribute values before computing correlations.
- *Demographics* specifies those attributes (e.g. *age*) in the demographic part of a user model that can be used for finding groups of similar users (see Section 4.2).
- *Interests* mirror the domain taxonomy of the Deep Map database. In its current version, the domain taxonomy covers seven areas of interest (namely *Restaurants, Buildings, History, Art, Nature, Sports*,

and *Economy*). The interest sub-tree comprises five levels with nearly 500 leaf entries. Figure 7 shows a small portion of the domain taxonomy. *Environmental Burden* is currently selected in the left frame and its attributes are shown in the right frame. The attribute value for *classifier* specifies the classification template to be used for discretizing interest probabilities for *Environmental Burden*. The attributes *mentor_finding* and *mentor_prediction* control whether *Environmental Burden* should be included in the mentor finding process and whether predictions based on similar users should be computed for this interest. Both flags are on for *Environmental Burden*.

3.1.4. *Service model*

Each entry in the Service Model represents a description of a server-internal event in which a user modeling component is interested. Event subscriptions specify the type of LDAP request that should be monitored (e.g. an add operation) and reported to one or more user modeling components (e.g. MLC and DIC). So-called "basefilters" allow one to restrict the portion of the overall taxonomy that must be monitored (e.g. merely *DMI Events*). Events can be triggered before and after an LDAP operation is executed by the server. Post-notifications allow a user modeling component to react to the outcome of an LDAP operation (e.g. start processing an interface event that has been added to *DMI Events*). Pre-notifications allow a user modeling component to be invoked beforehand (e.g. carrying out consistency checks on interface events that have been added to *DMI Events*). The following LDAP operations can be tracked by the Scheduler before and after execution: bind, unbind, search, modify, add, delete, rename, compare, and abandon. User modeling operations that are implemented as extensions to the LDAP protocol (e.g. for creating and deleting user models) can also be monitored and communicated to user modeling components.

3.2. *Communication*

Communication between external UMS clients and the Directory Component is possible via three different interfaces, namely $FIPA_{DM}$, *LDAP*, and *ODBC* (see Figure 3). In the following sub-sections, we briefly describe each of these interfaces. Within the UMS, an additional level of communication is based on the Common Object Request Broker Architecture (CORBA, Pope 1997; OMG 2001). In Figure 3, the CORBA Object Request Broker (ORB) is depicted on the right side of the Directory Component as a software bus that mediates between the Directory Component and the User Modeling Components.[8] For more information on this additional level of communication, we refer to Section 3.3.

3.2.1. *$FIPA_{DM}$ interface*[9]
Autonomous software agents that communicate via high-level messages significantly reduce the coupling between different modules (as opposed to the rather tight coupling between modules when employing established component frameworks like COM and RMI). An agent-based architecture hence facilitates evolutionary software development and was therefore adopted in Deep Map. The main aim of the $FIPA_{DM}$ interface is to mediate between Deep Map's messageoriented communication framework and the functionally structured LDAP interface of the UMS. Messages from Deep Map modules to the UMS are translated into one or more calls to the LDAP interface of the UMS and vice versa. Proactive communication can also be established by the UMS (e.g. the UMS alerts WebGuide that a user's interest in buildings has significantly increased).

3.2.2. *LDAP interface*
Native LDAP connectivity is provided by the directory server. In order to ease model management (e.g. creation and deletion of user models), several extensions to the LDAP protocol have been implemented using LDAP's extended operation facilities. The rationale behind this was to relieve administrators and applications from laborious and error-prone tasks (e.g. creating the initial topography for new user models, setting appropriate default access control rights, and populating new user models with default assumptions). Another reason was to centralize the execution of those management tasks that are critical for model consistency (e.g. new user models should become correctly initialized even in the case of software and hardware errors).

3.2.3. *ODBC interface*
The ODBC interface to the UMS allows one to (i) define relational mappings for the hierarchical LDAP representation used within the UMS, (ii) access these mappings from a variety of applications via ODBC, and (iii) periodically monitor user model content for certain conditions (e.g. creation of new user models, change of user model content). Appropriate actions can be invoked in such a case (e.g. a re-computation of user group models in a background process when user model contents change). Due to the wide support for ODBC on Windows-based software platforms, the ODBC interface also enables many desktop applications and a variety of data analysis and visualization tools to directly access the models hosted by the UMS (see the discussion of the External Clients in Section 5).

3.3. *Scheduler*

The main task of the *Scheduler* is to mediate between Directory Server and the user modeling components. User modeling components can subscribe to certain types of UMS events by maintaining event subscriptions in the Service Model (see Section 3.1.4). This limits communication traffic and allows one to freely add or remove user modeling components at runtime, and even to dynamically migrate them across a network of computers. After the launch of the UMS, the Scheduler loads event subscriptions from the Service Model. Subsequently, the Scheduler supervises LDAP events within the UMS and communicates these events and associated data to user modeling components that match their subscriptions.

A second task of the Scheduler is the provision of LDAP-compliant user modeling functionality (e.g. for creating and deleting user models) since there is no adequate support for these tasks in the standard LDAP protocol. In the case of, e.g. the creation of a new user model, this comprises the execution of several standard LDAP operations in a particular order, namely (i) checking for an already existing model, (ii) establishing the basic topography of a new model, (iii) setting appropriate access rights, and (iv) populating the model with default values. Moreover, rollback mechanisms have to be provided that preserve model consistency in case of potential problems during the creation process. Centralizing these administration tasks in the Scheduler solves many consistency problems and relieves administrators and application programmers from laborious and error-prone administration and programming tasks.

As an example, assume that the Scheduler communicates the addition of a usage event to the ULC. Figure 8 gives an overview of the consequences for the most important components, namely the directory server, the Scheduler, and the ULC:

- A user's request for a hypermedia document results in the communication of an event vector to the UMS that includes the term *Environmental Burden*.[10] The vector is inserted into the Usage Model via an LDAP add operation. An event subscription that the ULC had stored in the Service Model requests the Scheduler to communicate add operations after their execution by the Directory Server. In observance of this, the event vector is first stored in the *DMI Events* part of the Usage Model (see Section 3.1.2).
- Subsequently, the add event is handed over to the Scheduler, which starts scanning its internal subscription tables for matching entries. The event type and the basefilter of the ULC subscription match the current event; hence the Scheduler prepares this event for communication to

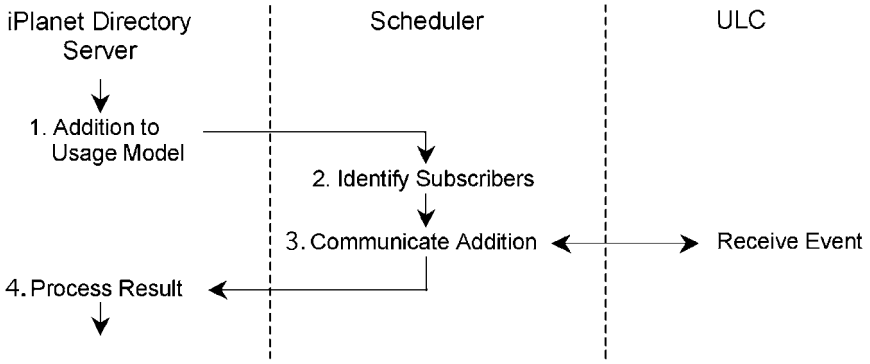

Figure 8. Scheduling scenario.

the ULC (in the case of several matching subscriptions, the event is communicated to all subscribers).

- The Scheduler communicates the add event and associated data (mainly the event vector) to the ULC, thereby following the processing specification contained in the subscription.
- The Scheduler reports the successful completion of all operations to the directory server, and controls the further processing of the LDAP operation (e.g. in the case of communication problems between the Scheduler and the ULC, the Scheduler can instruct the directory server to communicate this problem to the client that invoked the initial operation).

4. User Modeling Components

On the right side of Figure 3, we find the user modeling components that are currently implemented in the UMS for Deep Map (other components could be added as desired):

- The *User Learning Component* (ULC) learns user interests and preferences from usage data, and updates individual user models.
- The *Mentor Learning Component* (MLC) predicts missing values in individual user models from models of similar users.
- The *Domain Inferences Component* (DIC) infers interests and preferences in individual user models by applying domain inferences to assumptions that were explicitly provided by users or implicitly acquired by the ULC and the MLC.

We will describe each of these components in more detail in the following sub-sections.

4.1. *User learning*

A very active strand of user modeling research at present is devoted to learning users' interests based on relevant features of objects that users have viewed, rated, put in electronic shopping carts, bought, etc. The resulting assessment is often used for recommending and filtering objects (cf. Oard 1997). This kind of filtering is called *feature-based* filtering (or sometimes *content-based filtering*),[11] since it is based on object features. For instance, assume that users' interest in movies is determined by movie features like genre, actors, director, etc. A learning algorithm would then attempt to learn a user's preferences with respect to these features based on his or her interaction history, and thereafter rate new movies as to whether or not they are presumably interesting to this user. The result can be exploited, e.g. for recommending movies that the user will presumably rate highly, and for supplying and emphasizing features about movies that are presumably relevant for the respective user.

4.1.1. *Prior work*

Plenty of work has already been carried out in the area of learning about users' interests. An early system by (Jennings and Higuchi 1993) employed neural networks for filtering Internet news before presenting them to users. Other work includes Fab (Balabanovic 1997; Balabanovic and Shoham 1997), Letizia (Lieberman 1995) and Labour (Pohl et al. 1999; Schwab and Pohl 1999; Schwab et al. 2000). While these early systems used a single technique for a specific learning task, some recent research evaluated the application of multiple techniques to the same information filtering tasks (Pazzani and Billsus 1997; Breese et al. 1998; Herlocker et al. 1999). In Syskill and Webert (Pazzani and Billsus 1997), for example, several machine learning techniques (namely the nearest-neighbor algorithm and its PEBLS variant (Cost and Salzberg 1993), induction of decision trees with algorithms like ID3 (Quinlan 1986), two neural network approaches, and the naive Bayesian classifier (Duda and Hart 1973) were evaluated regarding their performance and accuracy in acquiring interest profiles from explicit user ratings on a set of biomedical documents. Based on this evaluation, the naive Bayesian classifier was chosen as the default algorithm for Syskill and Webert. Further improvements regarding learning accuracy have been achieved by emphasizing those features that are particularly relevant for classifying biomedical documents (e.g. by letting the user select and rate the importance of document features for classification).

Many of the algorithms discussed in the literature (e.g. those evaluated for Syskill and Webert) had to be discarded in this project due to their reliance on negative evidences of user interest. Users are known for not giving very

much feedback on the appropriateness of presented items, particularly not negative feedback (see the discussion in Schwab et al. 2000; Schwab and Kobsa 2002). Some systems like Letizia (Lieberman 1995) compensate this by regarding those options (namely web links) that users did not select as evidences of their non-interest. This however seems hardly appropriate for web-based systems. If a user clicks on some links on a web page but not on others, this does not imply that the other links are not interesting to him. This is especially true in cases where a web page does not fit on a computer screen, e.g. due to screen size, resolution, or excessive page length. There is evidence in the hypermedia literature that the parts of a hypermedia page that do not fit on a physical screen (and thus require scrolling to be seen) are only viewed by a relatively small number of users (Nielsen 1996). For all these reasons, we decided to use learning algorithms that solely rely on positive evidences of user interest.

4.1.2. *Method*

Of those algorithms that have been developed and evaluated in the literature to determine whether a user is interested in specific object features, we selected the univariate significance analysis (cf. Mitchell 1997; Schwab and Pohl 1999). The main reason for this choice is the ability of this algorithm to

- learn incrementally, which allows for keeping up with users' changing interests and preferences,
- represent learning results explicitly, which allows for leveraging synergetic effects between different learning methods explicitly,
- employ a domain taxonomy, which can considerable improve the learning process (both in terms of computational complexity and the quality of the learning results),
- rely on positive learning examples only, which relieves WebGuide from having to request or infer negative evidences of user interest, and
- demonstrate scalability, e.g. when the number of users, system's usage, and document features increase.

Univariate significance analysis is a statistical technique that is based on the assumption that the occurrence of identical object features in the navigation histories of users is normally distributed (see Figure 9).[12] If a feature appears in an individual user's navigation history less frequently than in a random sample, the user can be considered not to be interested in it. If a feature appears more frequently than in a random sample, the user can be assumed to be interested in this feature. In order to determine those non-interests and interests that are statistically significant, we introduce two confidence limits c_l and c_u for the lower and upper limit, respectively. If the actual number of feature occurrences in a user's navigation history is above c_u or below c_l,

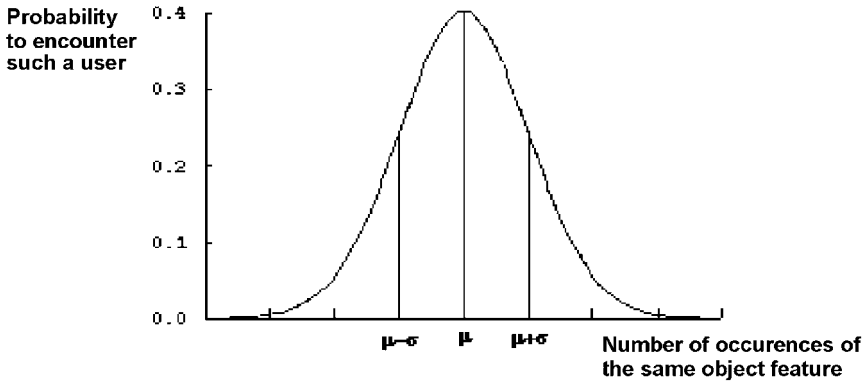

Figure 9. Normal distribution of users' interest in a given object feature.

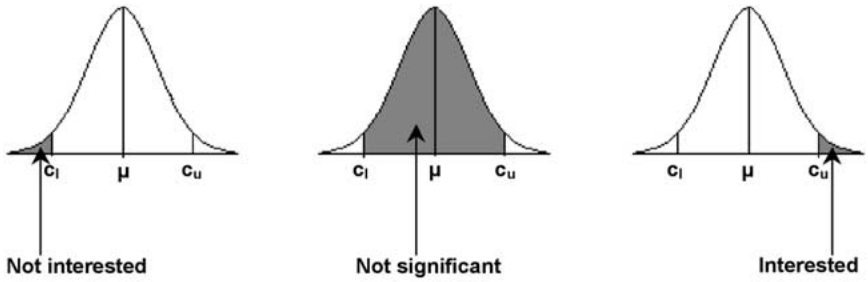

Figure 10. Classification of a user's interest.

we classify him as being interested in this feature or as not being interested, respectively. If the value is between the two limits, then no assumption is justified. The three cases are depicted in Figure 10.

ULC maintains the following assumptions about a user's interest in an object feature:

- the probability of the user's interest based on feature occurrences in his navigation history (henceforth called *individual probability* p_i),
- the probability of the user's interest based on feature occurrences in his navigation history in relation the occurrences distribution for all users (henceforth called *normalized probability* p_n), and
- a *classification* whether the user can be considered to be significantly interested in an object feature or whether this assumption is not justified.

4.1.3. *Usage scenario*

In the following scenario, we want to determine whether and to what extent *Peter Smith* is interested in the feature *Environmental Burden*. WebGuide sends event vectors containing this feature to the UMS, e.g. whenever a user

requests documents that discuss environmental burden in Heidelberg. We further assume that the UMS already collected in its Usage Model n = 216 occurrences of *Environmental Burden* for all users, and a total of N = 715 occurrences of all features for all users. Thus, the probability p to randomly select this feature is

$$p = \frac{n}{N} = \frac{216}{715} = 0.3 \tag{1}$$

We further assume that WebGuide reported N_a = 30 event vectors for Peter Smith and that n_a = 15 of them contained the feature *Environmental Burden*; hence, the individual probability p_i for this feature being contained in Peter Smith's event vectors is

$$p_i = \frac{n_a}{N_a} = \frac{15}{30} = 0.5 \tag{2}$$

Based on this, the ULC calculates the normalized probability p_n for Peter Smith as follows:

$$p_n = \frac{1}{1 + e - 0.4 \cdot \frac{n_a - p \cdot N_a}{\sqrt{p \cdot (1-p) \cdot N_a}}} = \frac{1}{1 + e - 0.4 \cdot \frac{15 - 0.3 \cdot 30}{\sqrt{0.3 \cdot 0.7 \cdot 30}}} = 0.72 \tag{3}$$

In order to determine whether or not a user's interest is statistically significant, we introduce two confidence limits c_l and c_u:

$$c_{l/u} = \mu \mp z \cdot \sqrt{p \cdot (1-p) \cdot N_a} \tag{4}$$

μ is the means of the distribution and equals the overall probability p multiplied by the total number of users' event vectors ($p \cdot N_a$), while z is the critical value. It determines the area under the standard normal distribution that falls within the confidence interval. For a confidence rate of 95%, the critical value z is 1.96. This means that 95% of random samples fall within this interval and 5% lie outside. In order to increase the confidence, we have to increase z accordingly (e.g. for a confidence of 99%, z is 2.576).

The ULC calculates the 95% confidence limits for classifying Peter Smith's interest in *Environmental Burden* as follows:

$$c_l = 0.3 \cdot 30 - 1.96 \cdot \sqrt{0.3 \cdot 0.7 \cdot 30} = 4.08 \tag{5}$$

$$c_u = 0.3 \cdot +1.96 \cdot \sqrt{0.3 \cdot .07 \cdot 30} = 13.92 \tag{6}$$

After rounding, c_l and c_u can be interpreted as follows with respect to Peter Smith's interest in *Environmental Burden*:

- If there are 4 or fewer events with this feature, he can be considered to have no interest in it.

- If the number of events is 14 or above, he can be considered to be interested in this feature (this is the case in our example with $n_a = 15$).
- Otherwise, we cannot assume a significant interest or non-interest of him.

After computing these probabilities and classifying Peter Smith's interest, the ULC checks whether his user model already contains an entry for *Environmental Burden* and whether the attributes individual_probability, normalized_probability, and classification are present. If the entry *Environmental Burden* exists, the ULC checks whether Peter Smith (or an administrator on behalf of him) recently inserted or modified this entry, as reported by the attributes *creatorsname* and *modifiersname* (see Section 3.1.1). Based on the outcome of these checks, the ULC proceeds as follows:

- If there is no entry for *Environmental Burden*, the ULC inserts it. If necessary, the ULC also inserts the superordinate concepts from the domain taxonomy (e.g. the more general interest in *Nature*, see Figure 5).
- If *Peter Smith* (or an administrator on behalf of him) recently inserted or modified assumptions about his interest in *Environmental Burden*, an update of these assumptions is not performed. The rationale behind this is that the ULC gives higher priority to user-initiated modifications than to system-initiated ones (a similar prioritization can be found, e.g. in (Kay 1995; Kobsa and Pohl 1995). Trust in a source is thereby given priority over recency of information when resolving the conflict between an existing user model entry and a newly acquired assumption.[13]
- If an entry about the user's interest in *Environmental Burden* is already present in the user model and does not strongly differ from the new system-generated value, the update (which is relatively costly in terms of system resources) is also not performed.

4.2. *Mentor learning*

A different approach to the prediction of unknown characteristics of the current user (particularly his interests and preferences) is to compare him with similar users. This approach has been called "collaborative filtering" (Goldberg et al. 1992) and more recently also "clique-based filtering" (Alspector et al. 1997; Kobsa et al. 2001).[14] The employment of this technique in our UMS compensates for some shortcomings of the approach pursued in the ULC, namely frequency analysis and classification of feature occurrence (see also Herlocker et al. 1999; Billsus and Pazzani 2000; Kobsa et al. 2001):

- The user must have been interacting with the user-adaptive system for a while before the ULC can determine user interests based on feature occurrence. This may be a critical restriction, e.g. in applications where

a user model is maintained during a single session only and in scenarios where users expect immediate personalized services in return for, e.g. granting permission that their personal data may be stored and processed for adaptation purposes.

- Content-based features may not be the main determinants for users' interest in objects. This is particularly true when users' personal tastes, esthetical judgements, etc. come into play.

4.2.1. *Prior work*

Early work on the inclusion of models of similar users into the user modeling process was carried out by, e.g. (Orwant 1995). His system "Doppelgänger" is a user modeling server that periodically (like every night) applies a clustering algorithm to models of individual users. Clusters found during this process are represented in models of user groups. As is the case with stereotype methods (see e.g. Rich 1979; Rich 1983; Rich 1989), information from group models can henceforth be employed when information in individual models is missing. An important difference to stereotype reasoning is that changes in individual user models can be taken into account by regularly re-applying the clustering procedure. (This is not likely to occur often since clustering is computationally expensive (Fisher 1987, 1996) and in practice needs to be supervised by experts who analyze the clusters and iteratively refine the clustering process). More recently, (Paliouras et al. 1999) followed a dual approach by integrating stereotypes that had been acquired in a supervised learning stage and clusters acquired in an unsupervised learning stage from individual user models. As opposed to (Fisher 1987), however, (Paliouras et al. 1999) did not report problems stemming from the computational complexity of their clustering processes.

A computationally more manageable and currently prevalent group of algorithms for clique-based filtering is a correlation-based neighborhood approach (see, e.g. (Resnick et al. (1994); Hill et al. (1995); Shardanand and Maes (1995); Konstan et al. (1997))). Its basic idea is to first select a subset of users that are similar to the current user based on known characteristics, and to subsequently compute predictions for unknown characteristics based on a weighed aggregate of their ratings. Alternative algorithms for clique-based filtering include Bayesian networks, vector-based similarity techniques, and induction rule learning (see Breese et al. 1998; Herlocker et al. 1999) for an overview and evaluation). Bayesian networks are reported to compute predictions faster and to have less resource requirements than correlation-based approaches. They require however a dedicated learning phase before they can be employed for clique-based filtering (i.e. they cannot learn incrementally in real time). Deployed to real-world scenarios, this implies that

the "learning phase (...) can take up to several hours and results in a lag before changed behavior is reflected in recommendations" (Breese et al. 1998). Such a system behavior seems inappropriate for the usage scenario of WebGuide (see Section 2.1) and hardly appropriate for many other real-world user modeling scenarios.

4.2.2. *Method*

We decided to employ *Spearman correlation* for determining the proximity between users in the MLC, which is based on ranks rather than values. Although this approach requires an initial classification step for several attribute types (e.g. characters, strings), it considerably increases at the same time the number of user characteristics that can be leveraged for finding similarities (including e.g. demographic attributes like sex and age). Another advantage of Spearman correlation is that it does not rely on model assumptions (e.g. regarding the probability distribution of attribute values), as opposed to the widely used Pearson correlation (Breese et al. 1998; Herlocker et al. 1999). The overall mentor learning process is then carried out in the following three steps.

a) Finding similar users

Similarity between two users is determined by computing the (linear) Spearman correlation coefficient for the two user models:

$$w_{a,u} = \frac{\sum_{i=1}^{m}(rank_{a,i} - \overline{rank_a})(rank_{u,i} - \overline{rank_u})}{\sqrt{\sum_{i=1}^{m}(rank_{a,i} - \overline{rank_a})^2}\sqrt{\sum_{i=1}^{m}(rank_{u,i} - \overline{rank_u})^2}} \qquad (7)$$

$w_{a,u}$ thereby is the similarity weight for the active user *a* and a neighbor *u* based on *m* assumptions that are available for both *a* and *u*. $rank_{a,i}$ represents the rank of the value of assumption *i* for user *a*. $\overline{rank_a}$ is the mean rank for all assumptions about user a. The transformation of values into ranks uses the classifiers that are contained in the System Model (see Section 3.1.3), thereby replacing values by class means.

b) Selecting mentors

Once the Spearman correlation coefficients have been computed for a given user, a relatively small number of most similar neighbors (the so-called *mentors*) must be selected from the set of similar users. In general, prediction accuracy increases with an increasing neighborhood. The increment in accuracy was however often found to decrease and even to turn negative at some point when adding more neighbors (Shardanand and Maes 1995; Herlocker et al. 1999; Sarwar et al. 2000). From a performance point of

view it also seems advisable to restrict the number of potential neighbors to a reasonable number in real-world environments with ten thousands of users.

The current version of the MLC can be configured to use correlation-thresholding or a best-n-neighbors approach when selecting mentors. Correlation-thresholding selects all those neighbors as mentors that have a correlation coefficient greater than a predefined threshold (Shardanand and Maes 1995). Depending on the situation, however, there may be too few or too many mentors remaining. The best-n-neighbors approach is to pick a fixed number of most similar neighbors as mentors. If there are only few highly correlated neighbors, this approach may select many neighbors as mentors whose correlation is low, thereby possibly reducing the prediction accuracy. In case of many highly correlated neighbors, neighbors may also be excluded that could have added to the accuracy of the prediction. The current version of the MLC for Deep Map uses the best-n-neighbors approach (with n = 20) as a default, which showed a good performance with regard to coverage and accuracy[15] in the empirical evaluation of (Herlocker et al. 1999).

c) Computing predictions

A variety of approaches have been discussed in the literature for computing predictions from a set of mentors. In MLC, an assumption i about the active user a is predicted using the following deviation-from-mean approach (Resnick et al. 1994; Konstan et al. 1997) over the selected mentors (n is the number of mentors, $w_{a,u}$ is the similarity weight for the active user a and a mentor u):

$$P_{a,i} = \overline{rank_a} + \frac{\sum_{u=1}^{n}(rank_{u,i} - \overline{rank_u}) \cdot w_{a,u}}{\sum_{u=1}^{n} w_{a,u}} \tag{8}$$

The deviation-from-means approach takes into account that users' means may vary. Therefore, u's ranks are normalized by their means, and the prediction for a is normalized by a's means. If no mentors were found for a given user, we compute the prediction as the average of the deviation-from-means across all users, that is,

$$P_{a,i} = \overline{rank_a} + \frac{\sum_{u=1}^{n}(rank_{u,i} - \overline{rank_u})}{n} \tag{9}$$

This approach exhibited a good coverage and accuracy in the evaluations by (Herlocker et al. 1999) and performed much better than the following simple means across all users:

$$P_{a,i} = \frac{\sum_{u=1}^{n} rank_{u,i}}{n} \tag{10}$$

In MLC, this simple means is only used when the User Model of the current user does not yet contain any assumptions at all (neither ones that were explicitly provided by the user, nor ones acquired by the ULC).

In summary, the method for clique-based filtering that is employed in MLC,

- allows for incremental learning and can thus keep up with a user's changing interests and preferences (cf. Section 4.1.2 for a discussion of the same quality in ULC),
- represents learning results explicitly (also cf. Section 4.1.2 for ULC),
- is able to adapt to a variety of user modeling needs (e.g. it allows designers to differentiate between mentors and stereotypes and to assign different weights, and can integrate algorithmic extensions like default voting, inverse user frequency, and case amplification (Breese et al. (1998))), and
- performs well in terms of predictive coverage and accuracy, responsiveness, and resource consumption.

4.2.3. *Usage scenario*

Table 1 shows part of the interest models of the fictitious users Joe, John and Al. Their models already contain many assumptions about the probability of their interest in different types of buildings in Heidelberg. This information has been either explicitly provided by them or acquired by the ULC from WebGuide usage data. During design time, the user model developer had also defined a stereotype "Art Lover" with some presumably typical characteristics of art lovers, and made it available to the MLC. At this point it was unclear though whether an interest in bridges can be attributed to this user group. Therefore, the designer decided to let the MLC predict this interest at runtime, based on the similarity of the stereotype to models of individual users. Another missing piece of information is the probability of John's interest in mansions.

Before computing the Spearman correlation, the MLC has to convert the interest probabilities into ranks, thereby utilizing the classifier for the respective interests in the System Model. Table 2 depicts the resulting interest models with associated mean ranks for the case in which the classifier divides the interval (0,1) into ten equal classes.

In order to find similar users, the MLC calculates the Spearman correlation coefficients for all pairs of users. The result is shown in Table 3.

The next task for the MLC is to select mentors for John and Art Lover from their neighborhoods. Applying a correlation threshold of 0.3 to the correlation coefficients, the MLC cannot find an appropriate mentor for John since Al's similarity seems too weak[16] and since Joe and Art Lover even exhibit interest

Table 1. Initial interest models.

Users	Interests			
	Churches	Restaurants	Mansions	Bridges
Joe	0.80	0.30	0.90	0.70
John	0.30	0.90	?	0.50
Al	0.30	0.40	0.70	0.30
Art Lover	0.90	0.10	0.90	?

Table 2. Initial interest models with ranked user interests.

Users	Interests				Mean
	Churches	Restaurants	Mansions	Bridges	
Joe	0.85	0.35	0.95	0.75	0.73
John	0.35	0.95	?	0.55	0.62
Al	0.35	0.45	0.75	0.35	0.48
Art Lover	0.95	0.15	0.95	?	0.68
Mean	0.63	0.48	0.88	0.55	

Table 3. Spearman correlation coefficients.

$W_{a,u}$	Joe	John	Al	Art Lover
Joe	1.000	–0.813	0.352	**0.986**
John		1.000	0.235	–0.882
Al			1.000	0.249
Art Lover				1.000

probabilities that are nearly the opposite of John's. For the stereotype Art Lover, the MLC identifies Joe as a very good mentor. Al and John are not considered, since their correlation coefficients are below the threshold.

Finally, the prediction algorithm can be selected and predictions can be computed. Since no mentor has been found for John, the MLC calculates the prediction for his interest in mansions as a deviation-from-mean average across all users. The interest in bridges for the stereotype Art Lover can be predicted as a deviation-from-mean with Joe as the only mentor. Table 4 shows the resulting complemented interest models.

Table 4. Interest models including predictions.

Users	Interests			
	Churches	Restaurants	Mansions	Bridges
Joe	0.80	0.30	0.90	0.70
John	0.30	0.90	**0.89**	0.50
Al	0.30	0.40	0.70	0.30
Art Lover	0.90	0.10	0.90	**0.68**

4.3. *Domain inferences*

4.3.1. *Method*

The Domain Inference Component (DIC) complements the user modeling functionality provided by the ULC and the MLC by applying domain-based inferences to individual user models. User interests that were explicitly provided by users or implicitly acquired by the ULC and MLC can give rise to additional assumptions about interests of this user by making inferences that are based on the hierarchical structure of interests (cf. Section 3.1.1):

- *Sidewards propagation*: if the user is interested in a minimum percentage *s* of direct sub-interests of a given interest, then the user is assumed to be also interested in the remaining subinterests, with a probability that equals the means of the probabilities of the current subinterests;
- *Upwards propagation*: if the user is interested in a minimum percentage *u* of direct sub-interests of a given interest, then the user is presumed to also hold this interest, with a probability that equals the means of the probabilities of the current sub-interests (cf. Kobsa et al. 1994).

By using domain inferences, individual user models can become populated more quickly with additional assumptions, and applications like WebGuide can come up with personalized information and services sooner. Compared with mentor learning, domain inferences also need relatively few resources.

4.3.2. *Usage scenario*

We briefly exemplify the domain inferences carried out by the DIC using a hypothetical scenario from WebGuide. We assume that the interest part of Nathan's user model comprises assumptions about his interests in buildings and architectural styles in Heidelberg. Based on Nathan's interaction with WebGuide (he previously browsed through several documents about buildings and monuments in Heidelberg), the ULC and the MLC acquired several normalized and predicted probabilities (abbrev. p_n and p_p) for Nathan's presumable interests. The initial state of his profile is depicted in Table 5.

Table 5. Initial state of Nathan's user model.

Nathan's interest in . . .

Architecture						
Buildings				Styles		
Churches	Restaurants	Mansions	Bridges	Gothic	Romanesque	Baroque
$p_n = 0.9$	$p_n = 0.1$		$p_n = 0.5$	$p_n = 0.3$	$p_n = 0.5$	
$p_p = 0.75$					$p_p = 0.75$	

The DIC, which is subscribed to additions by the MLC (see Section 3.3), now starts applying its domain inferences. This process is controlled by several configuration parameters in the Service Model, including the following ones:

- upwards propagation is enabled,
- sidewards propagation is enabled,
- a threshold of $s = 75\%$ is set for sidewards inferences,
- a threshold of $u = 60\%$ is set for upwards inferences,
- a weight of 100% is assigned to interest probabilities predicted by the ULC, and
- a weight of 70% is assigned to interest probabilities predicted by the MLC.

The percentage of direct sub-concepts of Buildings in which the user is interested is 75%, which is equal to s and greater than u. Since both upwards and sidewards inferences are enabled, the DIC calculates the probability of Nathan's interest (abbrev. p_i for inferred probability) in Buildings and Mansions as follows:

$$p_i = \frac{\frac{0.9+(0.75 \cdot 0.7)}{1+0.7} + 0.1 + 0.5}{3} = 0.48 \tag{11}$$

The percentage of direct sub-concepts of Styles in which the user is interested is 66%, which is smaller than s but greater than u. The DIC therefore does not compute a probability for Baroque style. Nathan's presumable interest in Styles in general is calculated as follows:

$$p_i = \frac{0.3 + \frac{0.5+(0.75 \cdot 0.7)}{1+0.7}}{2} = 0.45 \tag{12}$$

Finally, the percentage of direct sub-concepts of Architecture in which the user is interested is 100%. The DIC calculates the presumable interest in Architecture as a simple means of the inferred interests in Buildings and

Table 6. Final state of Nathan's user model.

Nathan's interest in . . .

Architecture $p_i = 0.47$						
Buildings $p_i = 0.48$				Styles $p_i = 0.45$		
Churches	Restaurants	Mansions	Bridges	Gothic	Romanesque	Baroque
$p_n = 0.9$	$p_n = 0.1$	$p_i = 0.48$	$p_n = 0.5$	$p_n = 0.3$	$p_n = 0.5$	
$p_p = 0.75$					$p_p = 0.75$	

Styles and updates the user model. Table 6 depicts the final state of Nathan's user model.

5. External Clients

External clients provide information about the user to the UMS, and retrieve current information about the user from the UMS. Currently, the following types of external clients are used in Deep Map or are considered to be added in the future:

a) Deep map agents
Deep Map Agents are autonomous software components that provide tour recommendations, analyze spoken input and generate speech output, interface the World Wide Web, etc. Deep Map Agents loosely adhere to the FIPA agent specifications, particularly to (FIPA (1998a); FIPA (1998b)).[17] They

- communicate to the UMS various data about the user's interaction with the system (e.g. user requests for information on *Environmental Burden* in Heidelberg, or gothic works of art in a specific church);
- communicate to the UMS various data about user characteristics (e.g. users' demographic data);
- query the UMS for user characteristics (e.g. retrieve from the model of the current user an assessment regarding his interest in information about the Early Middle Ages, or retrieve all interests and preferences with probability greater than 0.7); and
- query the UMS for information about the system environment (e.g. which user modeling components are currently available, or where Deep Map sub-systems are currently located).

b) LDAP browsers
System administrators carry out user model management tasks (e.g. configuring UMS components, assigning access control, maintaining assumptions in models of individual users) via an LDAP browser. Within its interface, standard LDAP operations (e.g. adding, updating, and removing entries) can be applied to the models hosted by the UMS. Additional tools can be used for administering the UMS, like for instance commercially available access control management tools (Baltimore 2001; Netegrity 2001).

c) LDAP-compliant applications
Users and applications that take advantage of the UMS can inspect user model contents (e.g. search for a specific user, list all individual interests) using a variety of LDAP-compliant applications. Numerous types of applications are already available, including web browsers, e-mail clients, standard LDAP browsers, HTML-based user interfaces, and server applications that cooperate with LDAP servers. Development kits for customs applications are available as well. For a product listing we refer to (Fink 2002).

d) Analysis and visualization tools
The collective content of the user profiles in the UMS constitutes a valuable source of information about characteristics of the whole user population. Numerous data mining (Woods and Kyral 1997) and data visualization tools (Card et al. 1999) have been developed during the past few years that facilitate the discovery of interesting patterns, regularities, outliers etc. in such data. While originally developed outside the user modeling community, many data mining tools have been recently applied to user modeling tasks as well, e.g. for the segmentation of customer bases using clustering algorithms, for the analysis of customers' shopping behavior based on past purchase data by learning decision rules, and for learning relevant characteristics of users who typically respond to certain advertisements by training neural networks (Woods and Kyral 1997). The ODBC interface of the Directory Component greatly facilitates the interaction with such tools.

We anticipate that in the future such tools will also be employed for assessing the adequacy of user modeling techniques. This can be accomplished, for example, by applying competing data mining algorithms to user model contents and evaluate their results with respect to key dimensions like performance, coverage, accuracy, sensitivity, and specificity (cf. Herlocker et al. 1999).

6. Access Control

Access control grants or rejects access to a user model, thereby enforcing a predetermined security and privacy policy. Access is granted and/or denied for *objects* (e.g. users, user groups, roles, applications, components of the user modeling server), which carry out specific *operations* (e.g. read, modify, delete) on *subjects* (e.g. models of individual users, models of user groups, single assumptions in these models). The process of granting access is sometimes also called "(positive) authorization". Over the past two decades, quite a few access control models have been proposed which authorize, e.g.,

- *explicitly* (e.g. through dedicated access control rules) or *implicitly* (e.g. through specialization from more general access control rules),
- *positively* (i.e. granting access) or *negatively* (i.e. denying access),
- *strongly* (i.e. an authorization cannot be overridden) or *weakly* (i.e. a general authorization can be overridden by a more specific authorization), and
- by *positive defaults* (i.e. authorization is granted if not explicitly denied) or *negative defaults* (i.e. authorization is denied if not explicitly granted).

For an overview of access control models, we refer e.g. to (Castano et al. 1995; Howes et al. 1999) and, more related to user modeling systems, to (Schreck 2001; Kobsa and Schreck 2002). Two of the academic user modeling servers developed so far provide basic support for access control. Doppelgänger (Orwant 1995), the older of the two systems, takes advantage of the Kerberos[18] system for authenticating objects and provides authorization based on access control lists. BGP-MS (Kobsa and Schreck 2002) relies on the more sophisticated SSL (Secure Sockets Layer) technology for authentication, signing, and encryption, and takes advantage of a more flexible role-based access control model for authorizing objects. Moreover, BGP-MS provides support for anonymity and pseudonymity. Facilities for auditing and resource control, however, are largely lacking in both server systems.

Access control grants anonymous and authenticated clients access to directory information. Apart from the requirements for an access control model defined in RFC 2820 (Stokes et al. 2000), there is currently no standard access control mechanism for LDAP.[19] In our implementation, we decided to take advantage of the access control model offered by iPlanet Directory Server (iPlanet 2001) since we employ this commercial server as a basis for our user modeling server (see Section 3.1). iPlanet Directory Server establishes access control via a set of access control lists. Each of these lists implements an access control rule and is usually attached to a directory entry via the special attribute *aci* (for Access Control Information). An ACI grants access to the directory entry to which it is attached, and to all entries beneath

it (i.e. its children). Its granularity can be very fine (if necessary, down to a single attribute of a single entry). In general, an ACI comprises the following specifications (for a more formal presentation, we refer to (iPlanet 2001)):

- the *directory resources* (i.e. *objects*) to which the ACI applies (e.g. entries in a particular subtree, entries that match a given search filter, specific entries, specific attributes),
- the *access rights* granted (e.g. read, write, search, compare, add, delete), and
- the *directory clients* (i.e. *subjects*) to which the ACI applies (e.g. specific users, anonymous users, non-anonymous (i.e. known) users, all members of a user group, users specified in attributes of object entries, users from a specific IP address or a specific domain, users accessing the directory during a period in time).

The default in UMS is that all users are denied access to any directory. Starting from this, ACIs implement access control rules that grant access as follows:

- The System Model may only be modified by administrators and inspected by user modeling components.
- The Usage Model may only be modified by WebGuide and those user modeling components that process usage data (e.g. the ULC). Moreover, administrators may inspect the Usage Model.
- The Service Model may only be modified by administrators and user modeling components.
- User models may only be modified by their respective owners, the user-adaptive application WebGuide and those user modeling components that manipulate user model contents (e.g. the ULC). Administrators are allowed to inspect user model contents. Anonymous users may perform the LDAP operations read, search and compare on all user model entries, except for the potentially sensitive demographic attributes within a user model.[20]
- Operational attributes (see Section 3.1) may only be modified by the LDAP server.

7. Related work

Research on separating user modeling functionality from user-adaptive application systems and incorporating this functionality into generic systems that can be customized to serve the needs of a wide range of user-adaptive applications has already lead to a number of prototypes (e.g. UMT (Brajnik and Tasso 1994), BGP-MS (Kobsa and Pohl 1995; Pohl 1998), DOPPELGÄNGER (Orwant 1995), TAGUS (Paiva and Self 1995), and um

(Kay 1995)). Highly-valued characteristics of these systems were generality (including domain independence), expressiveness for the representation of different kinds of complex assumptions about the user, and strong inferential abilities that would allow for complex forms of reasoning about the user. (Kobsa 2001a) surveys these and other academic systems and explains why the criteria that were applied in their development do not seem so important any more today (except for the quest for abstraction and reusability).

Group Lens (Net Perceptions 2001), LikeMinds (Macromedia 2001), Personalization Server (ATG 2001) and Frontmind (Manna 2001) represent recent commercial developments that aim at supporting customer relationship application on the web (Fink and Kobsa 2000; Kobsa et al. 2001). Characteristics that are important in such application scenarios include quick adaptation (since users' interaction with such applications is often very short), extensibility (to allow for the integration of bespoke or third-party user modeling methods), integratability of external user-related information, load balancing (to accommodate drastic load changes), fallback mechanisms (in case of breakdowns), transactional consistency (to avoid inconsistencies due to parallel read/write on the user model or abnormal process termination), and support for privacy. (Fink and Kobsa 2000) show however that current commercial systems often fail to meet these and other important criteria. Sections 2.1 and 2.2 explain that the same holds largely true for the requirements that were found to be important in the domain of personalized tour recommendations (see Fink 2002, for the complete requirements catalogue that is both based on experiences from real-world deployments and the scientific literature on generic user modeling systems).

Related work also exists with respect to systems that generate personalized suggestions for tourists. AVANTI (Fink et al. 1996, 1998) catered the presentation of information about touristic sites, hotels and vacation cottages to users' interests and to their physical impairments (particularly visual and motoric impariments). The system took advantage of the user modeling server BGP-MS (Kobsa and Pohl 1995; Pohl 1998) in which information about all users is stored and made accessible from a central point through the Internet. AVANTI uses rules and a domain hierarchy for drawing inferences about users. It was designed to be used and partly tested in public terminals, travel agencies, and users' homes.

The Cyberguide project (Abowd et al. 1997) produced a series of prototypes of a mobile, hand-held context-aware tour guide for visitors of a research lab. The systems only take information about the current location and orientation of the user into account, but applications that require rudimentary knowledge about the user (e.g. the languages that he speaks) are envisaged as well.

The ILEX system (Oberlander et al. 1997; Cox et al. 1999) produces "intelligent labels" for artwork that can be delivered as web pages, in an electronic gallery, or as synthesized speech in a physical gallery. The system takes the objects into account that the user has visited or that were mentioned to him before, and refers to them when generating labels.

Hyperaudio (Not et al. 1997; Petrelli et al. 1999) is an audio-based mobile information system for museum visitors. It adapts information based on what the user is assumed to know due to stereotypes that pertain to him, what the user has already seen and heard and seems interested in, as well as his preferred language style.

HIPPIE (Oppermann and Specht 1999; Specht and Oppermann 1999) also gives users personalized information about the artwork and the interior of a museum. It can be used at home for the preparation of and the debriefing after a museum visit, and as a mobile guide during a museum visit (in the latter case, users receive explanations through headphones, and orientation and awareness information through a handheld device). The information selected and presented to visitors reflects their location (at home or in front of some object of art), interests (as determined by the objects they physically visited or selected on the screen), the knowledge acquired so far, and their presentation preferences (as determined by the object attributes that they selected). The system uses rules for learning from user behavior and thereby exploits a domain hierarchy.

GUIDE (Cheverst et al. (2000b); Cheverst et al. (2002)) is a handheld electronic tourist guide that creates city tours which are tailored to the individual visitor. The system takes personal details into account (e.g. name, age, sex) as well as the user's interests (e.g. maritime, history, dietary preferences), his current location, the landmarks he already visited, and tours he followed. The system traces the visitor's page requests and updates his interest profile accordingly.

The overall aim of CRUMPET (Poslad et al. 2001) is the development and evaluation of tourism-related value-added services for nomadic users across fixed and mobile networks. The system is planned to be aware of users' current spatial context, and to adapt to this context as well as to users' domain-specific interests, preferences and interaction histories. While Deep Map and GUIDE were focused on a single city, CRUMPET is planned to be a generic system that accesses a range of local GIS servers for different cities as well as different service and content providers. Models of users will remain intact after a city visit and can be reactivated in a different city. In addition to recommending touristic sights, CRUMPET will advertise shops, restaurants, entertainment places and events, as well as information, reservation, booking and payment services. than databases.

As far as we can tell from the literature on the above-mentioned systems, it should be possible without major problems to employ the User Models of our user modeling server for representing the assumptions that these systems maintain about the user, and the Usage Model for recording the interaction history. Likewise, the Domain Inference Component and the interest hierarchy of our user modeling server can be employed for representing the inference rules and the domain hierarchy of AVANTI and HIPPIE. Hyperaudio's stereotypes can be expressed as well, and the neural network based learning mechanism that (Sarini and Strapparava 1998) propose for it could easily become another User Modeling Component. Presumably, the development of the user modeling components of these systems would be considerably facilitated by using the UMS, and the mentioned systems would also profit from the benefits described in Section 2.1.

8. Discussion and Conclusion

The generic user modeling system that has been presented in this paper is an open, standards-based, and platform-independent tool that provides essential user modeling services for deployment to real-world environments. Applications have a choice among a set of core techniques for drawing assumptions about users. By integrating these techniques into a single server, synergistic effects between them can be leveraged, thereby compensating for shortcomings of individual techniques (e.g. in cases of performance or scalability problems, or when data is sparse or not yet available). New techniques can be easily added to the server at any time.

While previous user modeling systems stored data about users in database and knowledge representation systems, the UMS employs a directory management system for this purpose. As is explained in more detail in (Fink 2002), this offers significant advantages with respect to the

- management and retrieval of (user-related) information, which is compliant to widely established standards;
- addition of new (user-related) information types to the set of pre-defined ones;
- distribution of information across a network (in a very broad fashion and on a very low level, if needed), which often leads to better performance, scalability, availability, and reliability of the user modeling service;
- replication of information, which may also enhance the performance and the availability of the service, and is particularly useful in mobile applications, where clients can fairly frequently become disconnected from the network; and the

- security of information, by providing facilities for authentication, signing, encryption, access control, auditing, and resource control.

(Shukla and Deshpande, 2000) report that databases do better than directories in performance when the number of entries that match a query and the total amount of data that is returned for a query increase. When the number of matching entries and the overall result set is small, however, directories exhibited a far better search performance in the authors' evaluation than databases. Such a situation is given in personalization applications, where only a small number of assumptions about users are normally needed for catering information to him. Directories are optimized for the processing of search operations whereas their update performance is considered less important. This characteristic is also well suited for user-adaptive applications, where a high search performance of a user modeling system is a must for dynamically personalizing web pages in real-time. Individual observations are normally not supposed to change the personalized behavior of a system immediately. Rather, individual observations are first collected and evaluated, so that delays within a user's session are normally acceptable.

This paper described the first prototypical application of the UMS in a mobile tourist guide. Commercial deployments to websites with several 100,000s of users (e.g. www.n24.de) are currently underway.

Notes

[1] Part of the research presented here was supported by a grant of the European Media Lab to GMD – German National Research Center for Information Technology.

[2] From (EML 1999); reprinted with permission. Recommended routes in the original screenshot have been emboldened for better reproduction.

[3] Figure 2a is courtesy of European Media Lab, Heidelberg, Germany. Figure 2b from (Coors et al. 2000) is courtesy of Fraunhofer IGD, Darmstadt, Germany. Figure 2c: is courtesy of Fraunhofer IGD (© Eicken Photographie).

[4] The importance of acquiring and maintaining selected demographic data is underlined by (Kaul 1999). Her empirical work on tourism in Heidelberg revealed that the interest of tourists in 17 out of 19 touristic activities significantly depends on their age, and that 14 out of 19 interests significantly depend on their continent of origin.

[5] All models are being viewed with the LDAP browser/editor "Globus Browser" (Gawor 1999).

[6] Operational attributes (e.g. modifytimestamp) are exclusively maintained by the LDAP server for administrative purposes (e.g. for tracking changes to attribute values).

[7] DMI is an abbreviation for "Deep Map Interface".

[8] In the current version of the UMS for Deep Map, we use the commercial ORB VisiBroker from Inprise (Inprise 2001).

[9] We use the term $FIPA_{DM}$ (for FIPA Deep Map) to indicate that Deep Map is largely compliant to (FIPA 1998a, 1998b), with some exceptions though. See (EML 1999, 2000) for a discussion.

[10] This term can be regarded as characterizing the content of the requested hypermedia page (there may be more than one descriptive term for a page). It could come from the HTML "description" and "keywords" tags, or it could have been selected using a term significance measure such as DF/ITF (Sparck Jones 1972).
[11] The term content-based filtering is mainly used in domains where the objects of interest are documents. In this case, the features are those terms of a document that are considered to be representative for the content of the document.
[12] We aim at investigating and evaluating other statistical distributions in the future. The Beta distribution seems to be especially promising since it allows for the approximation of a variety of probability distributions (e.g. linear, normal, exponential and parabolic) with a single stochastic model and only a few model parameters. An early user modeling system that employed the Beta distribution for modeling user interests was Doppelgänger (Orwant 1995).
[13] See (Fink 1999) for a discussion of related consistency problems and their impact on the overall usability of user-adaptive applications.
[14] The term "collaborative filtering" does not seem appropriate anymore due to the predominance of implicit acquisition techniques in more recent systems, and their limited or in most cases even non-existent support of user collaboration.
[15] Coverage indicates the percentage of characteristics across all users for which the system was able to produce predictions. The accuracy of predictions generated by the system can be assessed using measures e.g. from statistics (like mean absolute error and root mean square error) and decision-support (e.g. receiver operating characteristic).
[16] The correlation coefficient lies within the interval (–1, +1) and can be interpreted as follows: (i) a positive correlation coefficient indicates that high ranks for one user are associated with high ranks for the other user, and vice versa; (ii) a negative value indicates that high ranks for one user are associated with low ranks for the other user, and vice versa; and (iii) the closer the correlation coefficient is to 0, the weaker is the relationship.
[17] More precisely, Deep Map's agent-based communication framework shows minor extensions to, and omissions from, the corresponding FIPA standards, for a variety of reasons including performance and ease of programming (see EML 1999, 2000) for a more detailed discussion).
[18] Kerberos (Kohl and Neuman 1993) is an authentication service that is based on a private key system (sometimes also called "shared secret" system). SSL (Secure Sockets Layer) in contrast is based on a public key system (see Grant 1997; Smith 1997; Diffie and Landau 1998).
[19] Although the access control models offered by different LDAP servers share some commonalities, there are still many differences that impair their interoperability (e.g. when replicating and migrating access control information from one LDAP server to another).
[20] Such a weak access control policy seems only adequate for development purposes and will need to be restricted in a deployed version with "real" users data.

References

Abowd, G. D., Atkeson, C. G., Hong, J., Long, S., Kooper, R. & Pinkerton, M. (1997). Cyberguide: A Mobile Context-Aware Tour Guide. *Wireless Networks* **3**: 421–433.

Alspector, J., Kolcz, A. & Karunanithi, N. (1997). Feature-Based and Clique-Based User Models for Movie Selection: A Comparative Study. *User Modeling and User-Adapted Interaction* **7**(4): 279–304.

ATG (2001). ATG Dynamo e-Business Platform, Art Technology Group. http://www.atg.com/products/.

Balabanovic, M. (1997). An Adaptive Web Page Recommendation Service. *First International Conference on Autonomous Agents*, 378–385. Marina del Rey, CA.

Balabanovic, M. & Shoham, Y. (1997). Fab: Content-Based, Collaborative Recommendation. *Communications of the ACM* **40**(3): 66–72.

Baltimore (2001). Baltimore Select Access, Baltimore Technologies. http://www.baltimore.com/selectaccess/index.html.

Bigfoot (2001). Bigfoot International. http://www.bigfoot.com.

Billsus, D. & Pazzani, M. J. (2000). User Modeling for Adaptive News Access. *User Modeling and User-Adapted Interaction* **10**(2–3): 147–180.

Brajnik, G. & Tasso, C. (1994). A Shell for Developing Non-monotonic User Modeling Systems. *International Journal of Human-Computer Studies* **40**: 31–62.

Breese, J., Heckerman, D. & Kadie, C. (1998). Empirical Analysis of Predictive Algorithms for Collaborative Filtering. *Proc. of the Fourteenth Annual Conference on Uncertainty in Artificial Intelligence (UAI-98)*, 43–52. San Francisco, Morgan Kaufmann.

Carberry, S. (ed.) (2000). *User Modeling and User-Adapted Interaction* **10**(2–3). Special Issue on Deployed User Modeling. Dordrecht, Netherlands, Kluwer Academic Publishers.

Card, S. K., Mackinlay, J. D. & Shneiderman, B. (1999). Information Visualization. In: Card, S. K., Mackinlay J. D. & Shneiderman, B. (eds.) *Readings in Information Visualization: Using Vision to Think*, 1–34. San Francisco, CA, Morgan Kaufmann.

Carroll, J. M. (ed.) (1995). *Scenario-based Design: Envisioning Work and Technology in System Development*. New York, NY, John Wiley and Sons.

Carroll, J. M. (ed.) (2000). *Making Use: Scenario-based Design of Human-Computer Interactions*. Cambridge, MA, MIT Press.

Castano, S., Fugini, M., Martella, G. & Samarati, P. (1995). *Database Security*. Reading, MA, AddisonWesley.

Chadwick, D. W. (1996). *Understanding X.500: The Directory*. London, Thomson.

Cheverst, K., Davies, N., Mitchell, K., Friday, A. & Efstratiou, C. (2000a). *Developing a Context-aware Electronic Tourist Guide: Some Issues and Experiences, CHI 2000*, ACH Conference on Human Factors in Computing Systems, 17–24. The Hague, Netherlands. http://www.comp.lancs.ac.uk/computing/users/kc/Papers/CHI-cheverst.pdf.

Cheverst, K., Davies, N., Mitchell, K. & Smith, P. (2000b). *Providing Tailored (Context-Aware) Information to City Visitors. International Conference on Adaptive Hypermedia and Adaptive Webbased Systems, AH 2000*, 20–31. Trento, Italy. http://link.springer.de/link/service/series/0558/papers/1892/18920073.pdf.

Cheverst, K., Mitchell, K. & Davies, N. (2002). The Role of Adaptive Hypermedia within a Context-Aware Tourist GUIDE. *Communications of the ACM*, May 2002.

Coors, V., Kray, C. & Porzel, R. (2000). *Zu komplexen Diensten mit einfachen natürlich-sprachlichen Interaktionen*. First International Workshop on Digital Storytelling, Darmstadt, Germany, ZGDV. http://www.eml.villa-bosch.de/english/staff/homes/porzel%20publications/15-DISTEL.pdf.

Cost, S. & Salzberg, S. (1993). A weighted nearest neighbor algorithm for learning with symbolic features. *Machine Learning* **10**: 57–78.

Cox, R., O'Donnell, M. & Oberlander, J. (1999). *Dynamic versus Static Hypermedia in Museum Education: an Evaluation of ILEX, the Intelligent Labelling Explorer*. Artificial Intelligence in Education Conference, Le Mans, France.

Deep Map (2001). *Deep Map: Intelligent, Mobile, Multi-Media and Full of Knowledge* (Project Homepage). European Media Laboratory. http://www.eml.org/english/research/deepmap/deepmap.html.

Duda, R. & Hart, P. (1973). *Pattern Classification and Scene Analysis*. New York, NY, Wiley and Sons.

EML (1999). *Annual Report 1998/1999*. European Media Laboratory, Heidelberg, Germany.

EML (2000). *EML Annual Report 2000*. European Media Laboratory, Heidelberg, Germany. http://www.eml.villa-bosch.de/deutsch/aktuell/EML_(2000).pdf.

Fink, J. (1999). Transactional Consistency in User Modeling Systems. In: Kay, J. (ed.) *UM99 User Modeling: Proceedings of the Seventh International Conference*, 191–200. Wien New York, Springer-Verlag. http://www.um.org.

Fink, J. (2002). *User Modeling Servers: Requirements, Design, and Implementation*. Ph.D. Thesis, Dept. of Mathematics and Computer Science, University of Essen, Germany (to appear).

Fink, J. & Kobsa, A. (2000). A Review and Analysis of Commercial User Modeling Servers for Personalization on the World Wide Web. *User Modeling and User-Adapted Interaction* **10**(2–3): 209–249.

Fink, J., Kobsa, A. & Jaceniak, I. (1997). Individualisierung von Benutzerschnittstellen mit Hilfe von Datenchips für Personalisierungsinformation. *GMD-Spiegel* **1**: 16–17. http://www.ics.uci.edu/∼kobsa/papers/1997-GMD-kobsa.pdf.

Fink, J., Kobsa, A. & Nill, A. (1996). *User-oriented Adaptivity and Adaptability in the AVANTI Project. Conference. Designing for the Web: Empirical Studies*. Redmond, WA. http://www.ics.uci.edu/∼kobsa/papers/1996-designing-web-kobsa.pdf.

Fink, J., Kobsa, A. & Nill, A. (1998). Adaptable and Adaptive Information Provision for All Users, Including Disabled and Elderly People. *The New Review of Hypermedia and Multimedia* **4**: 163–188. http://www.ics.uci.edu/∼kobsa/papers/1998-NRHM-kobsa.pdf.

FIPA (1998a). *FIPA 98 Specification Part 1: Agent Management, Foundation for Intelligent Physical Agents (FIPA)*. Geneva, Switzerland. http://www.fipa.org.

FIPA (1998b). *FIPA 98 Specification Part 8: Human Agent Interaction, Foundation for Intelligent Physical Agents (FIPA)*. Geneva, Switzerland. http://www.fipa.org.

Fisher, D. (1996). Iterative Optimization and Simplification of Hierarchical Clusterings. *Journal of Artificial Intelligence Research* **4**: 147–179.

Fisher, D. H. (1987). Knowledge Acquisition Via Incremental Conceptual Clustering. *Machine Learning* **2**(2): 139–172.

Gawor, J. (1999). *LDAP Browser/Editor*. http://www-unix.mcs.anl.gov/∼gawor/ldap.

Goldberg, D., Nichols, D., Oki, B. M. & Terry, D. (1992). Using Collaborative Filtering to Weave an Information Tapestry. *Communications of the ACM* **35**(12): 61–70.

Grant, G. (1997). *Understanding Digital Signatures: Establishing Trust over the Internet and Other Networks*. New York, NY, McGraw-Hill.

Herlocker, J., Konstan, J., Borchers, A. & Riedl, J. (1999). An Algorithmic Framework for Performing Collaborative Filtering. *Proc. of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, 230–237. New York. http://www.cs.umn.edu/Research/GroupLens/algs.pdf.

Hill, W., Stead, L., Rosenstein, M. & Furnas, G. (1995). *Recommending and evaluating choices in a virtual community of use. ACM CHI'95 Conference on Human Factors in Computing Systems*, 194–201. New York, NY.

Howes, T., Smith, M. & Good, G. (1999). *Understanding and Deploying LDAP Directory Services*. Indianapolis, IN, Macmillan.

ILEX (1998). ILEX Virtual Gallerie 2.0. http://www.cstr.ed.ac.uk/cgi-bin/ilex.cgi.

Inprise (2001). Inprise. www.inprise.com.

iPlanet (2001). iPlanet Directory Server. iPlanet. http://www.iplanet.com/products/infrastructure/dir_security/dir_srvr.

ITU-T (1993). X.500 Part 1 Overview of Concepts, Models and Services, ISO/IEC Standard 9594.

Jennings, A. & Higuchi, H. (1993). A User Model Neural Network for a Personal News Service. *User Modeling and User-Adapted Interaction* **3**(1): 1–25.

Kaul, E. (1999). Soziokulturelle Kategorisierung der Touristen in Heidelberg, Master Thesis, University of Heidelberg, Heidelberg, Germany.

Kay, J. (1995). The um Toolkit for Reusable, Long Term User Models. *User Modeling and User-Adapted Interaction* **4**(3): 149–196.

Kobsa, A. (2001a). Generic User Modeling Systems. *User Modeling and User-Adapted Interaction* **11**(1–2): 49–63.

Kobsa, A. (ed.) (2001b). *User Modeling and User-Adapted Interaction* **11**(1–2). Ten Year Anniversary Issue. Dordrecht, Netherlands, Kluwer Academic Publishers. http://umuai.informatik.uni-essen.de/anniversary.html.

Kobsa, A., Koenemann, J. & Pohl, W. (2001). Personalized Hypermedia Presentation Techniques for Improving Customer Relationships. *The Knowledge Engineering Review* **16**(2): 111–155. http://www.ics.uci.edu/∼kobsa/papers/2001-KER-kobsa.pdf.

Kobsa, A., Müller D. & Nill, A. (1994). KN-AHS: An Adaptive Hypertext Client of the User Modeling System BGP-MS. *Proceedings of the Fourth International Conference on User Modeling*, 99–105. Hyannis, MA. http://www.ics.uci.edu/∼kobsa/papers/1994-UM94-kobsa.pdf.

Kobsa, A. & Pohl, W. (1995). The BGP-MS User Modeling System. *User Modeling and User-Adapted Interaction* **4**(2): 59–106.

Kobsa, A. & Schreck, J. (2002). Privacy through Pseudonymity in User-Adaptive Systems. *Submitted.*

Kohl, J. & Neuman, C. (1993). The Kerberos Network Authentication Service (Version 5).

Konstan, J. A., Miller, B. N., Maltz, D., Herlocker, J. L., Gordon, L. R. & Riedl, J. (1997). GroupLens: Applying Collaborative Filtering to Usenet News. *Communications of the ACM* **40**(3): 77–87.

Lieberman, H. (1995). *Letizia: An Agent that Assists Web Browsing. Fourteenth International Joint Conference on Artificial Intelligence*, 924–929. Montreal, Canada.

Macromedia (2001). LikeMinds, Andromedia. http://www.macromedia.com/software/likeminds/.

Malaka, R. (1999). *Deep Map: The Multilingual Tourist Guide*. C-Star Workshop, Schwetzingen, Germany. http://www.eurescom.de/∼pub/fusenetd/Malaka.pdf.

Malaka, R. & Zipf, A. (2000). DEEP MAP – Challenging IT Research in the Framework of a Tourist Information System. In: Fesenmaier, D., Klein, S. & Buhalis, D. (eds.) *Informaton and Communication Technologies in Tourism 2000: Proceedings of ENTER 2000*, 15–27. Wien, New York, Springer.

Manna (2001). Frontmind. http://www.mannainc.com/products.html.

Microsoft (2001). Active Directory Architecture. http://www.microsoft.com/windows2000/techinfo/howitworks/activedirectory/adarch.asp.

Mitchell, T. (1997). *Machine Learning*. New York, NY, McGraw-Hill.

Net Perceptions (2001). Net Perceptions. http://www.netperceptions.com.

Netegrity (2001). Netegrity SiteMinder, Netegrity Inc. http://www.netegrity.com/products/index.cfm?leveltwo=SiteMinder.

Nielsen, J. (1996). Top Ten Mistakes in Web Design. http://www.useit.com/alertbox/9605.html.

Not, E., Petrelli, D., Sarini, M., Stock, O., Strapparava, C. & Zancanaro, M. (1998). Hypernavigation in the Physical Space: Adapting Presentations to the User and to the Situational Context. *The New Review of Hypermedia and Multimedia* **4**: 33–45.

Not, E., Petrelli, D., Stock, O., Strapparava, C. & Zancanaro, M. (1997). *Person-oriented Guided Visits in a Physical Museum. 4th International Conference on Hypermedia and Interactivity in Museums (ICHIM'97)*, 162–172. Paris, France.

Novell (2001). Novell NDS eDirectory. http://www.novell.com/products/nds.

Oard, D. W. (1997). The State of the Art in Text Filtering. *User Modeling and User-Adapted Interaction* **7**(3): 141–178.

Oberlander, J., Mellish, C., O'Donnell, M. & Knott, A. (1997). *Exploring a Gallery with Intelligent Labels. 4th International Conference on Hypermedia and Interactivity in Museums (ICHIM'97)*, 153–161. Paris, France.

OMG (2001). Object Management Group (OMG), OMG. http://www.omg.org.

Oppermann, R. & Specht, M. (1999). A Nomadic Information System for Adaptive Exhibition Guidance. *Archives and Museum Informatics* **13**(2): 127–138. http://fit.gmd.de/∼oppi/publications/NomadicInfoSystem.pdf.

Oppermann, R. & Specht, M. (2000). A Context-Sensitive Nomadic Information System as an Exhibition Guide. *Proceedings of the Handheld and Ubiquitous Computing Second International Symposium, HUC 2000*, 127–142. Bristol, UK.

Orwant, J. (1995). Heterogenous Learning in the Doppelänger User Modeling System. *User Modeling and User-Adapted Interaction* **4**(2): 107–130.

Paiva, A. & Self, J. (1995). TAGUS – A User and Learner Modeling Workbench. *User Modeling and User-Adapted Interaction* **4**(3): 197–226.

Paliouras, G., Karkaletsis, V., Papatheodorou, C. & Spyropoulos, C. (1999). Exploiting Learning Techniques for the Acquisition of User Stereotypes and Communities. In: Kay, J. (ed.) *UM99 User Modeling: Proceedings of the Seventh International Conference*, 169–178. Wien, New York, Springer-Verlag.

Pazzani, M. & Billsus, D. (1997). Learning and Revising User Profiles: The Identification of Interesting Web Sites. *Machine Learning* **27**: 313–331.

Petrelli, D., Not, E., Sarini, M., Stock, O., Strapparava, C. & Zancanaro, M. (1999). *HyperAudio: Location-Awareness + Adaptivity. CHI'99, Conference on Human Factors in Computing Systems*, 21–22. Extended Abstracts, Pittsburgh, PA.

Pohl, W. (1998). *Logic-Based Representation and Reasoning for User Modeling Shell Systems*. Sankt Augustin, Germany, infix.

Pohl, W., Schwab, I. & Koychev, I. (1999). *Learning About the User: A General Approach and Its Application. IJCAI'99 Workshop Learning About Users*. Stockholm, Sweden.

Pope, A. (1997). *The Corba Reference Guide: Understanding the Common Object Request Broker Architecture*. Sydney, Australia, Addison-Wesley.

Poslad, S., Laamanen, H., M. R., Nick, A., Buckle, P. & Zipf, A. (2001). *CRUMPET: Creation of Userfriendly Mobile Services Personalised for Tourism. 3G 2001 – Second International Conference on 3G Mobile Communication Technologies*, 26–29. London, England. http://www.emorphia.com/downloads/3g2001-crumpet.pdf.

Quinlan, J. R. (1986). Induction of Decision Trees. *Machine Learning* **1**(1): 81–106.

Reagle, J. & Cranor, L. (1999). The Platform for Privacy Preferences. *Communications of the ACM* **42**(2): 48–55.

Resnick, P., Iacovou, N., Sushak, M., Bergstrom, P. & Riedl, J. (1994). *GroupLens: An Open Architecture for Collaborative Filtering of Netnews. ACM Conference on Computer Supported Cooperative Work*, 175–186. Chapel Hill, NC.

Rich, E. (1979). *Building and Exploiting User Models*. Ph.D. Thesis, Carnegie-Mellon University, Pittsburgh, PA.

Rich, E. (1983). Users are Individuals: Individualizing User Models. *International Journal of Man-Machine Studies* **18**: 199–214.

Rich, E. (1989). Stereotypes and User Modeling. In: Kobsa, A. & Wahlster, W. (eds.) *User Models in Dialog Systems*, 35–51. Berlin, Heidelberg, Springer.

Sarini, M. & Strapparava, C. (1998). Building a User Model for a Museum Exploration and Information-Providing Adaptive System. *Proceedings of the 2nd Workshop on Adaptive Hypertext and Hypermedia at HYPERTEXT'98*. Pittsburgh, PA. http://ecate.itc.it:1024/strappa/hips-um-ad/hips-um-ad.html.

Sarwar, B. M., Karypis, G., Konstan, J. A. & Riedl, J. (2000). *Analysis of Recommender Algorithms for E-Commerce. ACM E-Commerce 2000 Conference*, 158–167. Minneapolis, MN.

Schreck, J. (2001). *Security and Privacy in User Models*. Ph.D. Thesis, Dept. of Mathematics and Computer Science, University of Essen, Germany. http://www.ics.uci.edu/~kobsa/phds/schreck.pdf. Revised version to appear with Kluwer Academic Publishers, Dordrecht, Netherlands.

Schwab, I & Kobsa, A. (2002). Adaptivity through Unobstrusive Learning. *K1* 4/02. Special Issue on Adaptivity and User Modeling (to appear).

Schwab, I., Kobsa, A. & Koychev, I. (2000). *Learning about Users from Observation. Adaptive User Interfaces: Papers from the 2000 AAAI Spring Symposium*, 102–106. Stanford, CA. AAAI Press. http://www.ics.uci.edu/~kobsa/papers/2000-AAAI-kobsa.pdf.

Schwab, I. & Pohl, W. (1999). *Learning Information Interest from Positive Examples*. UM99 Workshop on Machine Learning for User Modeling. Banff, Canada.

Shardanand, U. & Maes, P. (1995). Social Information Filtering: Algorithms for Automating 'Word of Mouth'. *Proceedings of CHI-95*, 210–217. Denver, CO.

Shukla, S. & Deshpande, A. (2000). LDAP Directory Services – Just Another Database Application? In: Weidong, C., Naughton, J. & Bernstein, P. (eds.) *Proceedings of the 2000 ACM SIGMOD International Conference on Management of Data*. New York, N.Y: ACM.

Smith, R. (1997). Internet Cryptography. Reading, MA, Addison-Wesley.

Sparck Jones, K. (1972). A Statistical Interpretation of Term Specificity and its Application to Retrieval. *Journal of Documentation* **28**: 11–21.

Specht, M. & Oppermann, R. (1999). User Modeling and Adaptivity in Nomadic Information Systems. *7th GI-Workshop Adaptivität und Benutzermodellierung in Interaktiven Softwaresystemen*, 325–328. Magdeburg, Germany. http://fit.gmd.de/~oppi/publications/ABIS-hip.pdf.

Stokes, E., Byrne, D., Blakley, B. & Behera, P. (2000). Access Control Requirements for LDAP. RFC 2820, IETF. http://www.ietf.org/rfc/rfc2820.txt?number=2820.

Wahlster, W. & Kobsa, A. (1989). User Models in Dialog Systems. In: Kobsa, A. & Wahlster, W. (eds.) *User Models in Dialog Systems*. Heidelberg – Berlin, Springer Verlag.

WebGuide (2001). WebGuide: A City Guide for the Internet, European Media Lab. http://www.eml.org/english/research/deepmap/deepgis/webguide.html.

Woods, E. & Kyral, E. (1997). Ovum Evaluates: Data Mining. Ovum, London, England.

Xybernaut (2001). Mobile Assistant IV Products Description, Xybernaut. http://www.xybernaut.com/product/prod_des.htm.