
User Modeling and User-Adapted Interaction **10:** 209–249, 2000.

A Review and Analysis of Commercial User Modeling Servers for Personalization on the World Wide Web[1]

JOSEF FINK* and ALFRED KOBSA†
*humanIT Human Information Technologies GmbH, Rathausallee 10, 53754 Sankt Augustin, Germany, E-mail: Josef.Fink@humanit.de
† Dept. of Information and Computer Science, University of California, Irvine, CA 92697-3425, U.S.A., E-mail: kobsa@uci.edu

(Received 21 January 2000; in final form 4 July 2000)

Abstract. The aim of this article is to present and discuss selected commercial user modeling systems against the background of deployment requirements in real-world environments. Following the recent trend towards personalization on the World Wide Web, these systems are mainly aimed at supporting e-commerce including customer relationship management. In order to guide and structure our review, we define a requirements catalogue that comprises the main dimensions of functionality, data acquisition, representation, extensibility and flexibility, integration of external user-related information, compliance with standards, concern for privacy, and system architecture. Apart from the novelty of such a comparison both inside and outside the classical user modeling literature, a presentation of the core features of these commercial systems may provide a source of information and inspiration for the design, implementation, and deployment of future user modeling systems in research and commercial environments.

Keywords: personalization, one-to-one marketing, customer relationship management, electronic commerce, deployment requirements, commercial user modeling, company profiles, product reviews, user modeling servers.

1. Introduction

In several application domains, user-adaptive software systems have already proved to be more effective and/or usable than non-adaptive systems. One of these classes of adaptive systems with clear user benefits are user-adaptive tutoring systems which were shown to often significantly improve the overall learning progress. These systems and their benefits have already been extensively reviewed in the user modeling literature (see e.g. most of the papers in Brusilovsky et al. (1998) and the evaluations in Eklund and Brusilovsky (1998); moreover, see Specht (1998) and Specht and Kobsa (1999)).

[1]The managing editor of this paper was Judy Kay, University of Sydney, Australia.

Less represented in the user modeling literature are user-adaptive (aka 'personalized'[2]) systems for e-commerce including customer relationship management. A few notable exceptions are Popp and Lödel (1996), Åberg and Shahmehri (1999), Ardissono and Goy (1999, 2000), and Jörding (2000). This is surprising since there already exists ample evidence for personalization going mainstream in e-commerce. Appian estimates that the revenues made by the online personalization industry, including custom development and independent consulting, will reach $1.3 billion in 2000, and $5.3 billion by 2003 (Appian, 2000a). Gartner predicts that "by 2003, nearly 85 percent of global 1,000 Web sites will use some form of personalization (0.7 probability)"[3] (Abrams et al., 1999). There are also many indications that personalization provides substantial benefits in this application domain as well (Hof et al., 1998; Bachem, 1999; Cooperstein et al., 1999; Hagen et al., 1999; Kobsa et al., 2001).

Utilizing personalization and the underlying 'one-to-one' marketing paradigm is of paramount importance for businesses in order to be successful in today's short-lived, complex, and highly competitive markets (Peppers and Rogers, 1993; 1997; Allen et al., 1998). One-to-one builds on the basic principles of knowing and remembering a customer and serving him as an individual. From a marketing point of view, traditional communication channels between a company and its customers continuously decrease in efficiency due to market saturation, product variety, and increasingly complex and autonomous behavior of clients with respect to goods (e.g., drivers of luxury cars can at the same time be regular customers at discount shops) and media (e.g., people use different media like television, newspapers and the Internet, sometimes even in parallel) (Bachem, 1999). Against this background, traditional user segmentations in marketing research with their inherent simplicity (e.g., customer behavior can be predicted from a few key characteristics), linearity (i.e., future customer behavior can be predicted from past behavior), and time invariance (i.e., market rules always apply) provide less and less useful information for adequate personalization and have to be complemented by the latest information about customers directly elicited from their (on-line) behaviors. Thereby, marketers expect to get more insights into the many facets of customer behavior which is often fairly complex, non-linear, and time-variant (Bachem, 1999; Cooperstein et al., 1999).

Forrester Research reports regularly about the personalization activities of selected e-commerce sites, for example in Hagen et al. (1999) about the efforts and resulting benefits of 54 U.S. sites. Allen et al. (1998) describe 29 personalized web sites.

[2]In e-commerce, 'personalization' is used as a generic term that denotes user-adaptive system features and user modeling issues as well. Despite its ambiguity, we will employ this term throughout our article because of its predominance in this area. In cases where it is necessary to refer to one of the two meanings, we will use well-established and more specific terms from user modeling research like 'adaptivity' and 'user modeling'.

[3]This follows the ranking of the world's best performing companies that is annually carried out by Business Week (Business Week, 1999).

Schafer et al. (1999) reviews the personalized web services and associated benefits of well-known e-commerce companies like Amazon.com, CDnow, eBay, Levis, E! Online, and Reel.com. Their web services leverage especially

(i) *action-to-item affinities* between user actions and product attributes (e.g., a system recommends books on science fiction based on users' past queries for books that belong to this category),
(ii) *item-to-item affinities* between products the user has already expressed an interest in and potentially relevant products (e.g., the system recommends buying recordable CD-ROM disks since the user put a CD-ROM writer into her shopping cart), and
(iii) *user-to-user affinities* between a user and like-minded users (e.g., the system recommends a book on programming languages because users with similar purchase behavior also bought books on this topic in the past).

In general, personalization has been reported to provide benefits throughout the customer life cycle including drawing new visitors, turning visitors into buyers, increasing revenues, increasing advertising efficiency, and improving customer retention rate and brand loyalty (Hof et al., 1998; Bachem, 1999; Cooperstein et al., 1999; Hagen et al., 1999; Schafer et al., 1999). Jupiter Communications reports that personalization at 25 consumer E-commerce sites increased the number of new customers by 47% in the first year, and revenues by 52% (Hof et al., 1988). Nielsen NetRatings (ICONOCAST, 1999) report that e-commerce sites offering personalized services convert significantly more visitors into buyers than e-commerce sites that do not offer personalized services. Although the research approach taken is not always transparent and/or satisfactory (e.g., regarding the methodology used and the conclusions drawn[4]), these figures indicate that personalization offers at least in part significant benefits[5]. Besides these evidences, there seems to be an even greater potential for personalization improving customer retention and brand loyalty. According to Peppers and Rogers (1993) and Reichheld (1996), improving customer retention and brand loyalty directly leads to increased profits because it is much cheaper to sell to existing customers than to acquire new ones (since the costs of selling to existing customers decrease over time and since the spending of loyal customers tends to accelerate and increase over time). Consequently,

[4]One problem for instance is that peronalization is hardly ever introduced in isolation on a web site, but in most cases together with other company measures that may also have an effect on the addressed benefits (e.g., marketing and promotion measures, improved customer service, improved site navigation, and reduced response times (Cooperstein et al., 1999)).
[5]Despite of these success figures, there is also evidence for poorly done personalization leading to lower customer retention, reduced profit margins, and lost sales (Hagen et al., 1999). The authors found a personalized drugstore that allows users to disclose allergy information, but recommended a drug that was unsuitable for people with the allergy that the user had entered. Affected users are likely to leave this web shop, possibly forever. Another example is a web store that presented an advertisement for a $19.95 surge protector to a user who had already put a $59.95 model in her shopping cart.

businesses today focus on retaining those customers with the highest customer life time value, on developing those customers with the most unrealized strategic life time value, and on realizing these profits with each customer individually (Cooperstein et al., 1999; Peppers et al., 1999).

In parallel to the advent of personalized e-commerce sites, numerous tool systems emerged during the last few years that aim at assisting companies in developing and deploying personalized web sites. The aim of this article is to present and discuss selected commercial systems that provide user modeling functionality, focusing on systems that are available as server products. We will refer to these systems as *user modeling servers*, i.e. centralized software components that offer their services to several applications in parallel. In general, very little information can be found about commercial user modeling servers (see Appian (2000b) for a notable exception). Developers and vendors are not particularly eager to supply concrete information to academic solicitors and, to the best of our knowledge, most commercial user modeling servers were not even mentioned so far in the user modeling literature.

The deployment-supporting features that are in part offered by these systems (e.g., performance, scalability in terms of user modeling workload, extensibility with respect to complementary user modeling methods, integration of pre-existing user information and domain knowledge, and privacy protection) seem to be of paramount importance in real-world environments. It is noteworthy that these core features contrast sharply with those characteristics that have so far been regarded as important in classical user modeling research[6], like generality, domain-independence, expressiveness and strong inferential capabilities. Therefore we expect that a presentation and discussion of these commercial systems and their features will provide a source of information and inspiration for further research and development of user modeling systems and user-adaptive applications. Moreover, even though it is true that these commercial systems have been considerably shaped by the marketing background described above, many of them also seem at least partially useful as tool systems for user modeling in research environments.

In the following section, we first motivate the centralized user modeling approach taken by many commercial user modeling servers. After that, we introduce a catalog of relevant requirements for structuring and guiding our review. We then present selected user modeling server products along the lines of this catalog, and compare and discuss the findings.

2. Centralized vs. Decentralized User Modeling

For exhibiting personalized behavior, software systems rely on a model of relevant user characteristics (e.g., interests, preferences, proficiencies, knowledge). Acqui-

[6]Particularly for the so-called 'user modeling shell systems' (see Kobsa (2001) for a review).

sition and management of these models is carried out by a dedicated user modeling component. Most of the research prototypes that have been developed so far follow a monolithic approach with the user modeling component being embedded in and becoming an integral part of the user-adaptive application (see for example Finin (1989), Brajnik and Tasso (1994), Kay (1995), and Weber and Specht (1997)). A parallel strand of research focused on centralized autonomous user modeling[7] and lead to the development of a comparatively small number of user modeling servers (see for example Kobsa and Pohl (1995), Orwant (1995), Konstan et al. (1997), Machado et al. (1999), and (Billsus and Pazzani, 2000)). In contrast with this, most current commercial user modeling systems have been designed as server systems right from the beginning. (A notable exception from this is 'Open Sesame!' (Caglayan et al., 1997), an interface agent that maintains all information about the user in an embedded user modeling component[8].)

Compared to embedded user modeling components, user modeling servers seem to provide promising advantages regarding their deployment, including the following ones (see also (Billsus and Pazzani, 2000)):

- *Up-to-date user information for holistic personalization.* Information about the user, his or her system usage, and the usage environment is maintained by a (central) user modeling server and put at the disposal of more than one application at the same time. Such a central repository of user information is in sharp contrast with the scattered and partially redundant modeling of user characteristics within today's applications (including those on the World Wide Web). One can assume that from a user's point of view, such a central repository will significantly contribute to a more consistent and coherent working environment comprising different user-adaptive applications.
- *Synergistic effects with respect to acquisition and usage.* User information acquired by one application can be employed by other applications and vice versa. Examples for such a scenario are different types of news readers (Resnick et al., 1994); news readers and personalized agents (Good et al., 1999); and various sensor applications, an e-mail filtering application and a personalized newspaper (Orwant, 1995). Acquisition and representation components of a user modeling server can be expected to take advantage of synergistic effects as well (Pohl and Nick, 1999).

[7]Centralized user modeling does not necessarily imply physical centralization of user-related information (although this has been the case in all research prototypes developed so far). A promising alternative seems to be the concept of virtually centralized user information (see Section 5).

[8]The reason for this 'abnormality' seems to be that Open Sesame! was originally released as a desktop learning agent (i.e., a user-adaptive application that incorporates user modeling functionality). More recently, the development of Open Sesame! has been abandoned in favor of the user modeling server Learn Sesame (Caglayan et al., 1997; Open Sesame, 2000). For more information on Open Sesame! and Learn Sesame we refer to Section 4.4.

- *Low redundancy with respect to application and domain independent information.* Information about, e.g., users' competence in handling computers, like the ability to manipulate interface elements within a WIMP (Windows, Icons, Menus, Pointer) interface, can be stored with low redundancy in a user modeling server (Fink et al., 1998) to make it available to all applications which they use.
- *Low redundancy* with respect to stereotypes and user group models. Information about user groups, either available a priori as stereotypes (e.g., (Rich, 1979; 1983; 1989; Paliouras et al., 1999)) or dynamically calculated as user group models (aka 'communities') (e.g. (Orwant, 1995; Paliouras et al., 1999)) can be maintained with low redundancy in a user modeling server.
- *Increased security.* Known and proven methods and tools for system security, identification, authentication, access control, and encryption can be applied for protecting user models in user modeling servers (Schreck, 2000).
- *Increased support for the holistic design, acquisition, and maintenance of user models.* In the past, many efforts in user modeling research have been devoted to user model representation and inference issues. In commercial settings, however, the main focus is on leveraging the potential of user-related information on an enterprise level, e.g. by improving customer retention rate and brand loyalty (Hagen et al., 1999). In this vein, areas of work include the

  (i) design of an enterprise-wide user model schema;
  (ii) development and communication of an appropriate privacy policy;
  (iii) acquisition of user-related information at every point of contact with the user throughout the enterprise (e.g., web site, retail, sales, customer service, direct marketing, call center);
  (iv) integration of complementary user information that is dispersed across the enterprise (e.g., demographic data from client databases, past purchase data from transactional systems, available user segmentations from marketing research, regularities in past purchase behavior found in data mining processes); and finally
  (v) provision of user information to different applications for personalization purposes.

Against this background, user modeling servers that allow for the (virtual) integration of existing information sources about users and enable access to information stored in user models, can provide the basic platform for such a personalization infrastructure (Truog et al., 1999; Fink, 2000).

In addition, many more general advantages of centralized systems design (e.g., centralized user modeling servers relieve clients from user modeling tasks and can take advantage of powerful hardware resources), as well as disadvantages (e.g., necessity of a network connection, potential central point of failure), also apply (see for example Goscinski (1991), Orfali et al. (1994), and Tanenbaum (1992)). A discussion must however be omitted here for reasons of brevity.

Despite these potential benefits of centralized user modeling, we believe that current and future usage scenarios for computing devices will require a more sophisticated architecture. These scenarios include

(i) *multi-computer usage* (e.g., of a PC at work, a laptop on the go, and a PC at home, whereby the latter two are only temporarily connected to a network),
(ii) *mobile computing*, where a user carries a small information device (e.g., a mobile phone, palmtop, or organizer) that can be temporarily connected to a network wherever she goes (access to a computer network, however, cannot always be guaranteed),
(iii) *ubiquitous information*, where a user conjures up her information environment at every point of interaction like information walls, information kiosks, and desktops, and
(iv) *smart appliances* like intelligent car control systems and household appliances like refrigerators that acquire and manage users' preferences.

These scenarios demand a personalization infrastructure that comprises centralized components (e.g., user modeling server), decentralized components (e.g., 'OPS' profiles[9], user model gatherers (Yimam and Kobsa, 2001), user modeling intermediaries between user modeling servers and adaptive applications), and user modeling components that are embedded into application systems (Bertram, 2000). Finding the right balance is an open field for future research.

3. Review Methodology

We organized our comparison of commercial user modeling servers as follows:

- *Development and validation of a requirements catalog* for guiding and structuring our review. In several validation rounds, we applied the catalog to the product information collected until then, in order to validate its applicability. Collection of pointers to products and companies from sources like
  (i) relevant portal web sites (e.g., Marketing 1to1 (2000), Personalization (2000), and 1to1web (2000)),
  (ii) consulting and research companies (e.g., Appian (2000a; b), Forrester (2000), Jupiter (2000), and Nielsen (2000)),
  (iii) producers' web sites (e.g., Autonomy (2000), Bowne (2000), Manna (2000a) and Net Perceptions (2000)), and
  (iv) online (business) magazines and newspapers.

[9]OPS (Open Profiling Standard) is a privacy standard proposed by Netscape, FireFly, and VeriSign that enables users to control the local storage and disclosure of their personal data to web applications (Reagle and Cranor, 1999).

Moreover, we screened the literature on online marketing and on user modeling for work on user-adaptive systems for e-commerce (see Section 1 for selected references).

- *Selection of a set of representative user modeling servers.* Our initial research revealed more than 50 products that provide personalization functions. Their scope of application, however, is very heterogeneous, ranging from generic web development and runtime environments to off-the-shelf customer relationship management systems. We therefore first focused on those products that solely offer user modeling functionality and are sold separately (following this, we had to discard, e.g. Broadvision's 'One-To-One' (Broadvision, 2000) and Microsoft's 'Site Server' (Microsoft, 2000), where the user modeling part is merely a small component in a comprehensive e-commerce application environment). We also required a certain minimum amount of documentation (whose quality even thereafter was poor in many cases). A second selection criterion was based on the similarity of products and the representativeness of a system for a whole similarity group based on its comprehensiveness and reputation.

  Based on the acquisition and inference methods employed by these systems, we decided to form two main groups: systems that implement '*collaborative filtering*' and systems that adhere to a *rule-based approach*. From those systems that can be associated with collaborative filtering (i.e., 'GroupLens' (Net Perceptions, 2000), 'Gustos' (Gustos, 2000), 'LikeMinds' (Macromedia, 2000), 'StoryServer' (Vignette, 2000)), we decided to elect *GroupLens* as the representative of the whole category. From the category of rule-based systems (i.e., 'Advisor Solutions Suite' (Blaze, 2000), 'FrontMind for Marketing' (Manna, 2000a), 'Personalization Server' (ATG, 2000)), we decided to stay with *Personalization Server* and *FrontMind*, due to their distinctive strengths and weaknesses. From those systems that we could not associate to one of these groups (i.e., 'Customer Management' (Blue Martini, 2000), 'Learn Sesame' (Open Sesame 2000), 'RightPoint' (RightPoint, 2000), 'SelectCast' (HNC, 2000)), we added *Learn Sesame* to our review list, mainly because of its sophisticated features and its status as an early pioneer of personalization.
- *Review and quality control.* After completing the reviews, producers were asked for their assistance in validating our findings (alas with little feedback), and cross-checks were made with other sources of information that were originally not utilized.

Below is the final version of the requirements catalog, which also features selected product data (for more information on some of the requirements we refer to Kobsa et al. (2001)):

I. *Company profile*, including name, place of business, and a brief history
II. *Product profile*, including

- *functionality* offered (i.e., acquisition of user-related information, user modeling, user-related adaptations)
- *data acquisition* including the input data, namely
  - user data (e.g., demographics, interests and preferences), and
  - usage data:
    - observable usage (e.g., selective actions, temporal viewing behavior, ratings, purchases, other confirmative and disconfirmative actions)
    - usage regularities (e.g., frequency, navigation patterns, situation-action-correlations)
    - environmental data (e.g., software platform, hardware platform, locale)
    - acquisition methods (e.g., acquisition rules, statistics, case-based reasoning, decision trees, neural networks, stereotype reasoning, group model reasoning)
- *representation* methods (e.g., attribute-value pairs, graph-based representations, production rules)
- *extensibility* and flexibility, especially with respect to complementary acquisition methods
- *integration of external user and usage information* and domain knowledge (e.g., from legacy systems and OPS profiles)
- *privacy* and compliance to existing or forthcoming standards (e.g., technical support for implementing standard privacy policies as defined for instance by the TRUSTe (2000) privacy branding program, compliance to OPS and 'P3P'[10]), and related technical implications (e.g., inspectability of user model contents)
- *architecture* (e.g., embeddable into applications, single-tier or multi-tier server component)
- supported *software and hardware* platforms and related APIs (Application Programmer Interface)
- *client base* and publicly accessible web sites that employ the product online

III. *Similar products* that are not being described in detail in this survey

4. Reviews

In the following, we will review selected user modeling servers that are available as standalone products. We thereby adhere to the requirements catalog that was presented in the previous section. In order to facilitate orientation, the requirements

[10]P3P (Platform for Privacy Preference Project) is a privacy framework that enables Web sites to communicate their privacy practices, and users to exercise their preferences over those practices. Although similar to OPS to some extent, the main difference is in their focus: the focal point of OPS is on the storage, disclosure, and transport of user data, whereas P3P focuses on the mediation of privacy practices, thereby allowing users and web services to reach an agreement and ensure that the facilitated release of data is in accordance with that agreement. P3P was proposed by the World Wide Web Consortium (W3C, 2000).

are repeated as run-in headings in italics within each review. The main sources of information used are referenced at the beginning of each sub-section.

4.1. GROUPLENS (NET PERCEPTIONS, 2000; APPIAN, 2000b)

Company. Net Perceptions has its roots in work on news filtering systems that started in 1992 at the University of Minnesota with the 'GroupLens' project (Resnick et al., 1994; Konstan et al., 1997). Group Lens provides users with Usenet news in which they are presumably interested. Postings of low quality (e.g., spams, nonsense) or low relevance are not proposed for reading. Automatic filtering of news is based on ratings of Usenet postings that have been provided by like-minded users. Such affinity groups are automatically built by the GroupLens system based on correlations in users' anonymously provided ratings of news articles. This approach has been called collaborative filtering and more recently also 'clique-based filtering' (cf. Alspector et al. (1997))[11]. In any case, the underlying filtering approach successfully automated the problem of finding like-minded people in the rapidly growing Usenet communities and providing personalized recommendations from their ratings history. For an overview and an evaluation of various filtering algorithms we refer to Herlocker et al. (1999) and Breese et al. (1998).

Product. After its foundation in 1996, Net Perceptions obtained an exclusive license for the GroupLens technology from the University of Minnesota. After a short technology transfer phase, a first prototype of the GroupLens product was released in November 1996. Since then, the GroupLens product has been developed considerably further and is currently available in version 4.0. Complementary products that take advantage of the GroupLens recommendation engine are available for e-commerce, knowledge management, online advertising, e-mail marketing campaigns, and for supporting call center personnel in providing personalized advice and suggestions to clients. In parallel to these commercial developments, work on related research issues is still carried out at the University of Minnesota. In the remainder of this section, we will focus on the GroupLens recommendation engine (sometimes also called Net Perceptions recommendation engine).

Functionality, data acquisition. The GroupLens product comprises a recommendation engine and an associated set of APIs (see Figure 1). Via this interface, applications can send ratings to, and receive predictions from, the GroupLens recommendation engine. Interest predictions are generated by GroupLens from ratings explicitly provided by users (e.g., in on-line forms), from implicit ratings derived from navigational data (e.g., products that the online customer viewed or put into the shopping cart), and from transaction

[11]In the remainder of this article we will use the term 'collaborative filtering' rather than the more appropriate term 'clique-based filtering', mainly because it is commonly used in commercial settings.

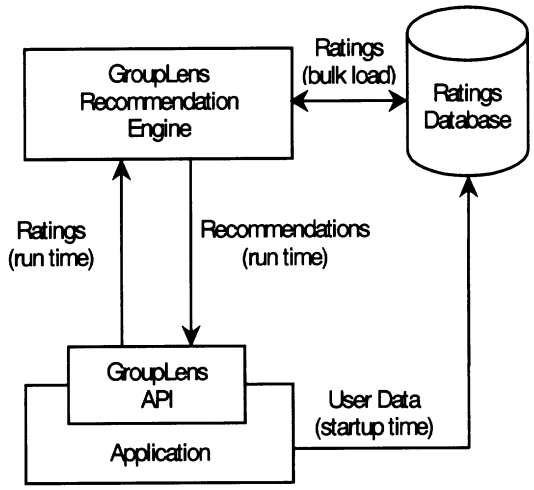

Figure 1. GroupLens Architecture (based on Net Perceptions (2000)).

history data (e.g., products purchased in the past). Whereas the first two types of user and usage information can be processed by GroupLens at runtime, past purchase data as well as past ratings can only be taken into consideration at startup time.

GroupLens offers three types of recommendations: personal, anonymous, and fast lookup (this corresponds to the affinity types introduced in Section 1: action-to-item, user-to-user, and item-to-item). Personal recommendations are calculated for a given user from her personal history of ratings and do not take ratings of other users into account (e.g., if a user rated science fiction films highly or purchased several SF films in the past, then GroupLens will recommend another SF film). Personal recommendations presuppose that the system has already been able to collect a meaningful amount of explicit and implicit user ratings that are indicators for a particular interest. If this is not the case, anonymous recommendations can be provided. In this case, a sample of already available user ratings is searched for similar users that correlate in their likes and dislikes with the current user. Based on this set of similar users, predictions about the probable ratings of the current user can henceforth be calculated. For those adaptations that require real-time response behavior (e.g., product up-selling or cross-selling recommendations) fast lookup predictions are provided by GroupLens since finding a group of similar users normally requires considerable computing resources in real-world environments with ten-thousands of users. Fast lookup predictions do not take users' ratings history into account but rely on predetermined relations between product attributes and categories. For instance, users who put a camera into their shopping cart will subsequently be offered recommendations on appropriate batteries and films.

Representation, integration of external user and usage information. In order to achieve real-time performance, GroupLens maintains all user-related data in cache memory. If this is not possible (e.g., due to shortage in memory), performance decreases significantly. GroupLens' cache memory is initialized from a ratings database, where all user ratings are stored. Applications that intend to communicate *a priori* available user data to GroupLens (such as purchase histories) have to convert them into an appropriate format and store them in the ratings database. After launching GroupLens, the database is initially sorted and subsequently loaded into memory. Depending on the data volume, this process is reported to take hours or even days. Besides this initial bulk loading facility for user data, no other technical means are provided for integrating legacy data at runtime. Figure 1 provides an overview of the GroupLens architecture and depicts the flow of information between the Recommendation Engine, the Ratings Database, and an Application.

Extensibility. GroupLens employs various collaborative filtering algorithms for generating predictions. Net Perceptions originally licensed algorithms from the University of Minnesota. During the last few years, Net Perceptions developed this technology further and now owns four pending U.S. patent applications. An example of these developments is a component called 'Dynamic Algorithm Selector', which chooses one out of several competing algorithms for generating predictions based on their performance and accuracy. Competing algorithms operate on the same tasks and run in parallel in the background. In general, performance is traded in for accuracy. Modifications and extensions of the algorithms in GroupLens by third parties are currently not supported and probably also not planned for the future. Besides collaborative filtering, Net Perceptions also claims to employ neural networks, fuzzy logic, genetic algorithms, and rule-based expert systems. However, these methods are not used in GroupLens but rather in the complementary product 'Ad Targeting'.

Privacy. Net Perceptions is actively contributing to and member of several privacy consortia (e.g. TRUSTe). In 1997, GroupLens was one of the first commercial user modeling products that supported the OPS proposal. In principle, GroupLens does not need to know the identity of a person in order to provide recommendations. In practice, however, most of the sites that employ GroupLens require a registration process, since they are not willing to offer the benefits of personalization without a payback in the form of personal data (a notable exception from this is the web site of CDnow).

Architecture, software and hardware. From an architectural point of view, GroupLens requires a dedicated server with a sufficient amount of memory (i.e., 128 MB or more recommended) and at least one processor that runs on Windows NT or Sun Solaris (IBM AIX support is forthcoming). A distribution of GroupLens across several computers on a computer network is not possible. Connection to a

web server or e-commerce server is established via an API that is based on CORBA[12]. Local interface support includes Java, C/C++, Perl, COM[13], and CGI[14]. Native database support is offered for Oracle and Microsoft SQL Server. ODBC[15] support that used to be provided by previous versions of GroupLens has been abandoned due to performance reasons.

Client base. Net Perceptions reports more than 190 customers as of March 2000. Web sites that employ GroupLens online include Bertelsmann-BOL, CDnow, E! Online, Kraft, and Ticketmaster Online.

Similar products. There are quite a few commercial systems on the market that can be associated with collaborative filtering. Among the products that are most similar to GroupLens are LikeMinds, and 'Firefly' which has been recently acquired and reportedly discontinued by Microsoft. In addition, there are several companies that hold patents regarding collaborative filtering and associated applications, including Microsoft, IBM, and AT&T. In general, the commercial interest in collaborative filtering seems to be quite high. Appian (2000b) reports that more than twenty U.S. patent applications explicitly discuss collaborative filtering and that nearly 100 describe various kinds of recommender systems. Relevant differences between GroupLens and LikeMinds include

(i) its modular architecture, comprising a front-end which generates various predictions based on a user's relation to a set of like-minded users, and a back-end which calculates sets of like-minded users;
(ii) its faculty for distributing the aforementioned back-end across a network of computers;
(iii) its recommendation engine that is separated into four isolated modules, each working on a different type of input data employed for recommendations (namely purchase data, navigational data, explicitly stated user preferences, and pre-defined product similarities); and

[12]CORBA (Common Object Request Broker Architecture) is an industry standard software platform for object communications. At the core of its architecture is the Object Request Broker (ORB), a component that enables software objects to communicate, irrespective of any physical details like location, hardware, operating system, and programming language. The CORBA standard is defined and propagated by the OMG (Object Management Group).
[13]COM (Component Object Model) is an object-oriented component software technology that is propagated by Microsoft. COM objects adhere to important object-oriented principles like encapsulation and reusability and can be written in a variety of programming languages including C++, Java, Visual Basic, and COBOL.
[14]CGI (Common Gateway Interface) is a standard programming interface for web servers. CGI-compliant programs extend the functionality of a web server and can generate customized web content dynamically.
[15]ODBC (Open Data Base Connectivity) is an industry-standard application programming interface for accessing relational data sources, including database management systems. In general, applications can access tabular data sources via ODBC, irrespective, for example, of the type of data management system used. ODBC has a wide third-party support due to Microsoft's strong commitment to ODBC on Windows-based software platforms.

(iv) its support for ODBC as a standard interface for database access (native database access is supported for selected database management systems, e.g. Oracle).

4.2. PERSONALIZATION SERVER (ATG, 2000; APPIAN, 2000b)

Company, product. Art Technology Group (abbr. 'ATG') was founded in 1991 as a consulting company by computer science graduates from MIT. At the end of 1998, ATG released its product Personalization Server as a complement to their previously released 'Application Server'. Application Server is the indispensable platform for all products from ATG and handles the processing of dynamic web pages on top of a web server. Personalization Server extends this functionality by profile management and a rule-based development and runtime personalization environment. Both products together provide the technical basis for complementary products from ATG, e.g. 'Commerce Server' for creating online shops and 'Ad Station' for providing promotions and advertisements. Figure 2 depicts the architecture of ATG's products suite and outlines some of the products' interdependencies (Commerce Server and Ad Station are not depicted). In the remainder of this section, we will mainly focus on Personalization Server, which is currently available in version 4.5.

Functionality, data acquisition, representation. Two main administrative applications are provided by Personalization Server: (i) a development environment

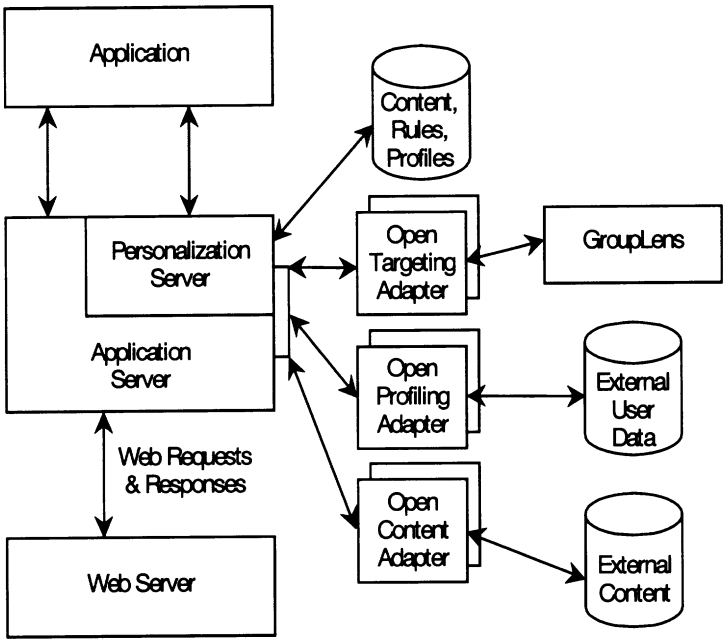

Figure 2. ATG Architecture (based on (ATG, 2000)).

called 'Developer Workbench', for creating personalized web pages; and (ii) an administration interface called 'Personalization Control Center', to allow non-technicians the definition of personalization rules and the maintenance of profiles. Up to and including the second quarter of 1999, ATG had no U.S. patents concerning their rule-based personalization technology.

Personalization Server's group profiles comprise relevant characteristics (e.g., age, gender) of user subgroups (e.g., family father, yuppie). Their development and maintenance has to be carried out manually. However, this work has normally not to be started from scratch, since existing user segmentations from marketing and feedback from Personalization Server's reporting facilities can at least serve as a basis. Rules that are associated with group profiles allow Personalization Server to assign an individual user to one or more user groups. These rules can take user data (e.g., demographic data like gender and age), implicit information about system usage (e.g., pages visited, products bought) as well as environmental information into account (e.g., domain name, browser type, operating system, available bandwidth). Most of the basic information used within rules is automatically captured by the system (e.g., information about the usage environment).

Figure 3 shows the user interface for defining user groups within Personalization Control Center. More specifically, it depicts the definition of a user group called 'High Risk Investors': all users who are presumed to have certain characteristics

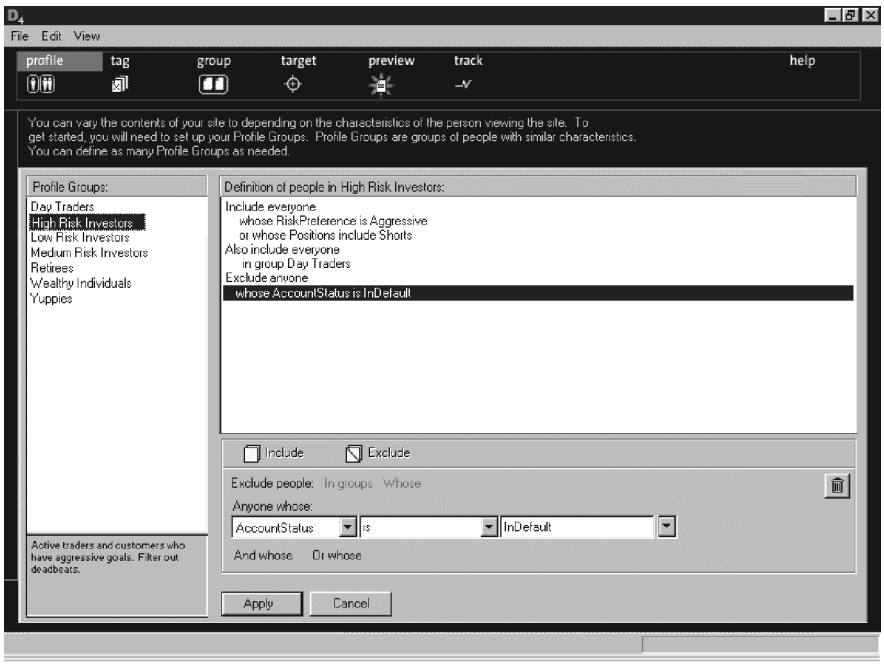

Figure 3. ATG Personalization Control Center (ATG, 2000). Reprinted with permission.

with respect to their 'Risk Preference', their 'Position', their membership to the user group 'Day Traders', and their 'AccountStatus', are assigned to this user group.

Group profiles and associated rules resemble very much the stereotype approach that was taken in many research systems (e.g., Rich, 1979; Moore and Paris, 1992; Kobsa et al., 1994; Ambrosini et al., 1997; Fink et al., 1998; Ardissono et al., 1999): a group profile forms the stereotype body (which comprises information about users to whom the stereotype typically applies), and one or more associated personalization rules function as activation conditions (or 'triggers') for assigning a group profile to an individual user.

Besides group profile activation, rules can also be employed for acquiring user information (e.g., ''When a person views Home Equity Loan Information, set DwellingStatus to HomeOwner'' (ATG, 2000)), and for providing personalized content (e.g., if a user is interested in football, then provide an article about the prospects of the forthcoming season). The editing and maintenance of rules is carried out via the Personalization Control Center, which puts a graphical user interface with several message windows and drop-down menus for defining dedicated rule parts at the disposal of the developer. Rule formats slightly differ depending on the intended purpose. Adaptation rules, for example, comprise the following information (for more details including a guided tour of Personalization Control Center we refer to ATG (2000)):

what; *who*; *when*(optional); *conditions* (optional)

What refers either to an information content category (e.g., football news) or to various selection conditions for information content (e.g., content items whose target audience are football fans). *Who* designates either a user or group profile, or contains one or more selection conditions on profile attributes (e.g., gender is male and income $>$ 60,000 and number-of-visits $>$ 3). The optional component *when* specifies a date for the rule becoming active. An arbitrary number of optional *conditions* forms the last part of a rule (e.g., whether the user's account is with AOL).

Besides this rather simple rule management interface, Personalization Control Center provides no further support for defining and maintaining rules. More sophisticated rule management features are e.g. offered by 'One-To-One' (BroadVision, 2000) and 'Advisor Solutions Suite' (Blaze, 2000). Examples of such features include (i) grouping of rules into 'rule packages' (sometimes also called 'rule sets'), (ii) a more fine-grained control over the life span of rules, and (iii) a simulation and testing environment for fielding a rule-based system. These mechanisms are indispensable when, for example, new rules are deployed into a real-world system that is driven by hundreds or thousands of personalization rules. The package mechanism then allows one to organize rules and restrict their scope to, for example, a single package. The life cycle control allows for the control of rules over time (e.g., by specifying an activation date, an expiration date, or a periodic activation). The simulation environment, finally, allows for the testing, logging, and tracing of the personalization system. Apart from this lack of more sophisticated rule management

features, another important drawback of Personalization Server seems to be that external user information (e.g., user segmentations from marketing databases, purchase information from transactional legacy systems) cannot be directly referenced in personalization rules.

Extensibility, integration of external user and usage information. Personalization Server offers interfaces called 'Open Targeting Adapters' (see Figure 2) for integrating complementary user modeling products and custom extensions into their rule-based user modeling approach (e.g., GroupLens from Net Perceptions). Moreover, Personalization Server provides interfaces called 'Open Content Adapters' and 'Open Profiling Adapters' (see Figure 2 for both adapter types) that allow for the integration of user information that is external to ATG's products (e.g., purchase data from legacy systems, user segmentations from marketing databases, demographic user data from customer databases). As mentioned above, incorporating such kinds of external information in personalization rules does however not seem to be envisaged in Personalization Control Center.

Privacy. During our research, we found no commitment of ATG to a dedicated privacy policy (e.g., for handling visitors' session data on their own web site). We also found no information about ATG's membership in (or active contribution to) privacy consortia, nor about Personalization Server's compliance to established privacy proposals in this area (e.g. OPS). A company White Paper (Singh, 2000) advocates the free collection (albeit not release or vending) of user behavior data. When comparing these efforts to those of their competitors (e.g., Net Perceptions, Bowne) it seems that ATG is rather insensitive with respect to privacy and is not very well able to support customers in developing an appropriate privacy policy and implementing appropriate technical means to enforce it.

Architecture. Personalization Server's architecture (which is backed by Application Server) supports server clusters and associated features like:

(i) *load balancing*: in order to maintain a consistent performance, new user sessions are delegated to available servers and, if none is available, new servers can be dynamically added to a server cluster,
(ii) *over-capacity contingency*: in case of traffic peaks, user requests are gradually delayed and finally rejected in order to prevent the whole system from crashing,
(iii) *request fail-over*: in case of a server crash, subsequent server requests are redirected to available servers within the cluster,
(iv) *session fail-over*: in case of a server crash, a user's session can be transparently migrated to another available server within the cluster without any loss of data, and
(v) *geographic fail-over*: if an entire server cluster becomes unavailable, subsequent server requests can be automatically redirected to another server cluster, probably running in a different geographic location.

Software and hardware. The deployment of Personalization Server to real-world settings seems to be supported very well. ATG's products are entirely written in

Java and run on top of Windows NT, Sun Solaris, and IBM AIX. For storing content, personalization rules and profile information about users and user groups (see Figure 2), all major JDBC[16]-compliant database management systems are supported including those from Oracle, Sybase, Informix, and Microsoft (for a detailed list of supported database management systems we refer to ATG (2000)).

Client base. ATG lists more than 150 customers on their web site up to and including the first quarter of 2000. BabyCenter, BMG Direct, living.com, and Newbridge Networks are some of the web sites that employ Personalization Server online. Other important customers include Sun Microsystems and Sony Networks.

Similar products. There are several commercial systems that are similar to Personalization Server, most notably One-To-One from BroadVision (2000) and StoryServer from Vignette (2000). In these systems, however, the user modeling components are embedded and form a rather small part of the overall product functionality (which additionally includes for example transaction handling, content management facilities for products, editorials, advertisements, and discussion groups). Relevant criteria that distinguish Personalization Server include:

(i) its Open Content and Open Profiling Adapters that allow the integration of external user-related information,
(ii) its scalable architecture that is based on server clusters, and
(iii) its compliance to industry standards like Java, Java Beans, Enterprise Java Beans, and Java Servlets (see SUN (2000) for an overview), which seems to support deployment to real-world settings very well.

Compared to Personalization Server, BroadVision's One-To-One offers increased support for user identification, authentication, and encryption through established standards like SSL, STTP, SET, X.509 digital certificates, and RSA encryption (see Schreck (2000) for an overview and a discussion of these standards with respect to user modeling). Moreover, One-To-One offers developers of user-adaptive applications sophisticated rule management and built-in support for user model inspection[17]. The acquisition methods offered by Vignette's StoryServer excel by complementing rule-based personalization with collaborative filtering. In order to achieve this, Vignette licensed a customized version of GroupLens (see Section 4.1).

[16]JDBC (Java Data Base Connectivity) is an industry-standard application programming interface for accessing relational data sources, including database management systems. In this aim, it is quite comparable to ODBC (see Footnote 15). The most important difference between the two is that JDBC caters solely to the needs of application programs written in Java. JDBC is propagated mainly by SUN Microsystems.

[17]According to Appian (2000b), however, it seems that many customers of BroadVision do not take advantage of this feature, since they do not want to put their clients in control of their profiles.

4.3. FRONTMIND (MANNA, 1999a,b; 2000a–d; APPIAN, 2000b)

Company, product. Manna was founded in 1997. The company is headquartered in Wellesley, Massachusetts, while research and development are located in Tel Aviv, Israel. Manna released its product 'FrontMind for Marketing' (abbr. FrontMind) in the first quarter of 1999, and version 2.0 in April 2000. FrontMind provides a rule-based development, management and simulation environment for personalized information and personalized services on the web. FrontMind distinguishes itself from other rule-based products like Personalization Server (see Section 4.2) by having Bayesian networks for modeling users' behavior integrated into their personalization framework. Manna owns a pending U.S. patent application for this technology. Although currently focused on personalizing web applications, efforts are underway to deploy FrontMind to call center, e-mail, and online direct marketing scenarios (see Section 1).

Functionality, data acquisition, representation. FrontMind comprises the following components (see Figure 4):

- *FrontMind Client* interfaces with web application servers and communicates user-related events (e.g., user logins, requests for information on a specific product) to, and receives adaptation recommendations (e.g., product cross-selling recommendations) from, the FrontMind Server.
- *FrontMind Server* provides a rule-based personalization environment that is based on the sub-components *Rule Evaluator* and *Learning and Inference Engine*. The Rule Evaluator receives (a configurable set of) user-related events from the FrontMind Client (see above) and matches them against those adaptation rules that are active at the time of evaluation. Information about users and products from the Clients & Products Database and dynamically acquired information about customers' behavior (see below) can be taken into account during the matching process. The first two sources of information are exploited in a deterministic way, whereas the latter one is used in a 'non-deterministic' (viz. probability and similarity based) manner. After having selected those rules that match all preconditions, their associated adaptation part is evaluated and the results are communicated back to the FrontMind Client. The Learning and Inference Engine acquires and maintains models of user behavior. Relevant dimensions in these models include referral web pages, system's usage (e.g., product information requests), purchase histories, and demographics (e.g., gender, age). Internally, these models are based on Bayesian networks. Selected parts of the domain (e.g., purchases of consumer electronics and professional products) can be represented in single models. If appropriate, these models can be arranged in a model hierarchy, thereby creating more comprehensive models (e.g., product purchases regarding appliances in general). Other model topographies (e.g., model chains) can be established as well (for more information on the agent-based background of these behavior models we refer to (Barnea, 1999)). According to Manna

(1999a; 1999b), designing and implementing behavior models is one of the main tasks that have to be accomplished when deploying FrontMind to customer sites. Besides employing behavior information for online adaptation purposes, FrontMind offers a variety of tools for offline model construction, analysis (e.g., unsupervised clustering for customer and market segmentations), simulation (e.g., 'what-if' decision support), and visualization (e.g., dependency graphs).

- '*Business Command Center*' (not depicted in Figure 4): an administration environment for non-technicians that allows for the definition and maintenance of adaptation rules and the simulation of their performance (e.g., matching rate) on the basis of historical data. An additional component reports the performance of business rules (e.g., their matching rate across user profiles, time periods, and product categories) and of the FrontMind system as a whole (e.g., page views, purchase patterns). The log files that support these simulation facilities are stored in the FrontMind Database along with the aforementioned adaptation rules.
- '*Business Object Developer*' (not depicted in Figure 4) is a tool that allows technicians the development of business objects, the basic building blocks that constitute every adaptation rule. Standard business objects are provided, e.g., for accessing the system's usage history, and users' profiles and behavior models. Internally, business objects can contain (i) database queries, (ii) calls to stored procedures hosted by database management systems, (iii) calls to the Learning

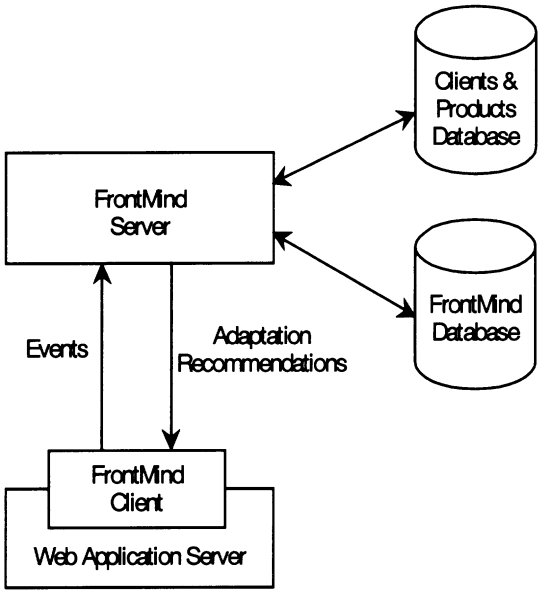

Figure 4. FrontMind Architecture (based on (Manna, 2000b)).

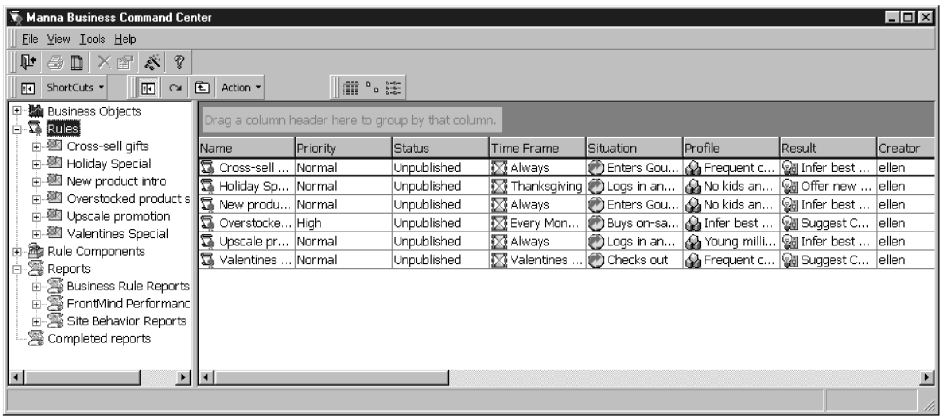

Figure 5. Manna Business Command Center (Manna, 2000b). Reprinted with permission.

and Inference Engine, and (iv) custom code written in Java for appropriately processing information from the aforementioned sources. Business objects are stored in the FrontMind Database.

- '*Console Manager*' (not depicted in Figure 4): administration tool for the FrontMind system that allows for the configuration of the FrontMind Server, the establishment of security policies, and for the tracking of system resource utilization.

The Business Command Center, whose main screen is shown in Figure 5, is the central administration interface for controlling personalization in FrontMind. The navigation tree in the left frame comprises the top-level sections Business Objects, Rules, Rule Components, Reports, and Completed Reports. In the remainder of this section, we will focus on Business Objects and Rules. For more information on FrontMind's reporting facilities we refer to Manna (2000a).

The section Business Objects comprises standard objects provided by FrontMind (e.g. 'Customer') and custom objects that may have been defined using the Business Object Developer. The Rules section, which is unfold, contains the adaptation rules that are currently defined in FrontMind. Rules details are shown in the right frame.

Time Frame denotes the times when a rule should become active (e.g., always, at a distinct day, for a period in time, on recurring dates and periods). *Situation* specifies those sets of usage events that trigger an adaptation rule (e.g., user login, information request for a particular product, when a product is put into the shopping cart). An adaptation rule gets activated by the Rule Evaluator only if the Situation matches the actual system's state. *Profile* denotes one or more conditions that refer to attributes in the current user's profile (e.g., whether the user's gender is female, whether the customer's lifetime value is high, whether the customer already ordered a particular product in the past) or to Business Objects that employ models of users' behavior (e.g., infer those user characteristics that describe potential buyers for a

particular product and check whether these characteristics match the current user's profile). *Result* determines the outcome of a rule, i.e. either a set of deterministic adaptations (e.g., show the advertisement for a particular product, propose complementary products for those already in the shopping cart) or non-deterministic adaptations that are (partially) driven by models of users' behavior (e.g., recommend the best-selling product for a particular category). The other rule attributes, *Name*, *Priority*, *Status* and *Creator*, should be self-explanatory.

Several major differences become apparent when comparing FrontMind with Personalization Server regarding data acquisition and representation:

- FrontMind maintains dynamic models of users' behavior, which can take arbitrary user and usage related information into account, whereas Personalization Server relies on rather static group profiles and associated acquisition and activation rules.
- FrontMind employs rules mainly for adaptation purposes[18], whereas Personalization Server also utilizes rules for acquiring assumptions about the user and for assigning profiles of user groups to individual users.
- Besides static user and usage related information, FrontMind's adaptation rules can also take advantage of users' behavior models.

Extensibility, integration of external user and usage information. FrontMind offers no dedicated high-level interfaces for integrating external user-related information (e.g., from a marketing database) and complementary user modeling products (e.g., recommendations from a system that relies on collaborative filtering). On the programming level, however, such external information sources can be quite organically integrated by encapsulating all necessary access details in Business Objects. Once this has been accomplished, these objects can henceforth be transparently employed in rules.

Privacy. Manna's privacy efforts are comparable to those of ATG: hardly any clear information about privacy policies, about compliance to established privacy proposals (e.g., OPS, P3P)[19], and about Manna's membership in or active contribution to privacy consortia. It seems that Manna does not consider privacy being a real challenge for their business and those of their customers;[20] hence, privacy efforts are rather limited and confined to a fairly general privacy statement.

[18]In all FrontMind brochures and White Papers that we had at our disposal, we found no example of a rule being employed for a purpose other than user-related adaptation, although this should be possible from a technical point of view.

[19]In Manna (2000b), we found the following statement: "The FrondMind Framework ... has a reliable and extremely secure server framework, which supports all major network security protocols, and privacy initiatives such as the Platform for Privacy Preferences (P3P) Project". Without any additional explanation and substantiation, this statement is not very satisfactory.

[20]The only comment on privacy we found was that "the main challenge e-businesses face today in implementing solutions is not addressing customer privacy issues (best handled with a privacy statement to customers affirming what data will be captured and how that data will be used)..." (see (Manna, 2000a) and (Manna, 2000c)).

Architecture. FrontMind relies on an agent-based communication and cooperation framework that allows for a very flexible management of software components, including their dynamic distribution across a network of computers (Manna, 2000b). FrontMind's exceptional flexibility is supported by a set of architectural features, including the following:

- *plug-ins*: are components of the FrontMind Server (e.g., Business Objects, adaptation rules) that can be created, executed, updated, and removed in real time without interrupting the server's operation,
- *load balancing*: automatically distributes user sessions according to resource requirements across a network of computers,
- *messaging*: all communication between software components is established via explicit messages that are represented in XML[21],
- *events*: the number and type of events that are communicated from the FrontMind Client to the server can be configured by the system administrator. All user actions that are at the disposal of the web application server can be selected to become input for FrontMind.

Software and hardware. FrontMind's sophisticated architecture and its Java-based infrastructure seem to support its deployment to real-world settings very well. The FrontMind Client can connect to web application servers via Active X[22], servlets[23], or sockets[24], depending on the operating system platform. The FrontMind Server can run on top of Windows NT and Sun Solaris. The minimum hardware configuration for the server comprises a dual processor board with at least 512 Mb RAM and 5 Gb of free disk space. On Solaris, the amount of memory should not fall short of 1 Gb. With regard to database management systems, FrontMind supports Oracle version 8.0 (or higher) and Microsoft SQL Server 7.0 (or higher).

Client base. Up to and including the first quarter of 2000, Manna reports six important customers on their web site (Manna, 2000a): Harcourt General, Saleoutlet.com, Get-Outdoors.com, Entertainment Boulevard, and GourmetMarket.com. There is no information available whether these companies really employ FrontMind online.

Similar products. For a list of products that are similar to FrontMind, we refer to the corresponding paragraph in Section 4.2. Features that set FrontMind apart

[21]XML (Extensible Markup Language) is a standardized markup language for the World Wide Web. Unlike HTML, XML allows for the definition of new markup elements within documents. Defining and applying custom markup elements to objects allows the communication of object semantics in addition to object content. XML is propagated by the World Wide Web Consortium.
[22]Active X is a software technology from Microsoft Corporation that facilitates the exchange of information between software components, applications, and web pages. Active X builds on another proprietary component technology from Microsoft called COM.
[23]Servlets are CGI-compliant programs that are written in Java.
[24]Sockets are a widely used application programming interface for remote inter-process communication via Internet protocols (i.e., TCP, UDP, IP) and other transport protocols (e.g., OSI TP/IP, X.25, DecNet, AppleTalk).

include: (i) the integration of dynamic models of users' behavior that are based on Bayesian networks into a rule-based approach, (ii) an agent-based communication and cooperation framework that allows for a flexible management of software components, and (iii) an interface for non-technicians (i.e., Business Command Center) that provides rule management, simulation, and reporting facilities.

4.4. LEARN SESAME (APPIAN, 2000b; BOWNE, 2000; OPEN SESAME, 2000)

Company, product. 'Open Sesame' was a former division of Charles River Analytics, which carried out contract research for various institutions over the last decade in the areas of intelligent agents, neural networks, and expert systems. Their first product 'Open Sesame!' was launched in 1993. Open Sesame! was a learning agent for the Macintosh platform that monitored a number of user action types at the interface level (e.g., opening and closing folders, documents, and applications) and proposed the automation of detected repetitive patterns (Caglayan et al., 1997). Patterns were found along two dimensions: time (e.g., a user's web browser is opened every day at 9:00 a.m.) and action sequences (e.g., the user usually launches a dictionary application before opening a text document).

Despite some successes of Open Sesame! (the company claims having shipped more than 35,000 copies), further developments aimed at significantly broadening the scope of deployment beyond a single hardware and software platform. The overall aim was to diversify Open Sesame! towards a user modeling server that could be easily integrated in applications, particularly in web applications. Besides architectural requirements, this objective challenged especially Open Sesame!'s learning functionality with issues like scalability, robustness, demand for efficient incremental learning, and controllability by client applications. Subsequent developments led to a new generation of learning algorithms, a corresponding U.S. patent application, and to a new product called Learn Sesame that was launched in 1997. During the second quarter of 1998, Open Sesame was acquired by Bowne Inc. The rationale behind this takeover was to strengthen Bowne's Consulting and Development branch by integrating Open Sesame's personalization experience. Today, Learn Sesame is available from Bowne both as a standalone product and as a component that is integrated into their Internet applications for the financial sector[25]. In the following, we will mainly focus on Learn Sesame (for more details of Open Sesame!, Learn Sesame, and a brief description of the transition from Open Sesame! to Learn Sesame, see Caglayan et al. (1997)).

Functionality, data acquisition, representation. Learn Sesame relies on applications for collecting implicit and explicit user, usage, and environmental data. Relevant usage characteristics (e.g., keywords of requested hypermedia pages,

[25] As of the time of writing, Bowne Inc. announced the sale of Open Sesame to Allaire Corporation (Allaire, 2000). Allaire will also take over the complete development team, including the original founder of Charles River Analytics. The consequences of this acquisition for the product are unclear as yet.

ratings of products, keywords entered in a search form) have to be collected by applications and sent to the user modeling server along with relevant user characteristics (e.g., user id, age, gender, sex, income). Learn Sesame analyzes this stream of time-stamped events for recurrent patterns, and supplies applications with evidences for regularities (e.g., a user's presumed interest in outdoor clothing, a correlation between the amount of money spent and suitable product categories, a correlation between product category and user demographics like age, gender, income, and formal education for a group of users).

Learn Sesame's learning algorithms are based on incremental hierarchical clustering[26]. Clusters of events and sequences are identified, analyzed, and so-called 'facts' are generated that describe relevant characteristics of clusters found. Examples include the number of events that 'support' (i.e. are in) a cluster, or the relative size of a cluster in comparison to the other clusters as an indicator of confidence. In this learning process, recent events have a higher impact on evidences for regularities than older ones. The process of identifying, analyzing, and communicating clusters can be controlled by applications. A few examples may illustrate this:

- A *timer interval* specifies the amount of time between subsequent clustering stages. Choosing a higher value results in less resource consumption but also in the generation of less timely facts, and vice versa.
- A *learning threshold* specifies the minimum number of events that are required for creating a cluster (see below for an example). Choosing a higher threshold results in the slower generation of more solid facts, whereas a lower threshold results in faster generation of less solid facts.
- A *confidence threshold* specifies the minimum confidence for a cluster to be communicated to applications (see below for an example). Choosing a higher value results in the communication of less facts with a higher confidence, whereas a lower value results in more facts with a lower confidence.

As already mentioned earlier, Learn Sesame provides support for incremental learning of user-related characteristics. Figure 6 depicts the six steps that make up every learning stage (the numbers used refer to those in the figure):

(1) An application generates events comprising user and usage data, e.g. the following five events about users' interest in products, each of them comprising a list of four attribute-value pairs:

UserId=12, ProductId=47, Price=Medium, Trendiness=Low
UserId=47, ProductId=97, Price=High, Trendiness=High
UserId=12, ProductId=17, Price=Medium, Trendiness=Low
UserId=12, ProductId=31, Price=Medium, Trendiness=Low
UserId=10, ProductId=99, Price=Low, Trendiness=Low

[26]In contrast, Open Sesame! used neural networks that were based on adaptive resonance theory (Caglayan and Snorrason, 1993; Snorrason and Caglayan, 1994).

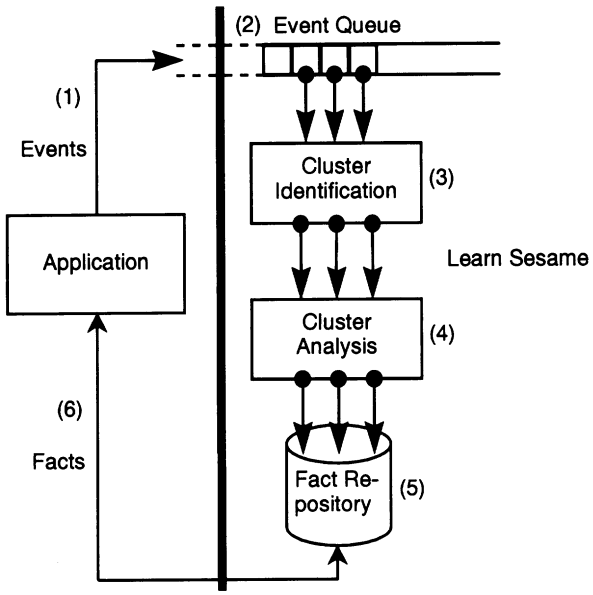

Figure 6. Incremental Learning Process (based on Caglayan et al. (1997)).

(2) Incoming events are buffered by Learn Sesame in an event queue, until the next clustering stage starts (which is controlled by *timer interval*).
(3) New clusters of events are identified and previously identified clusters are refined.
(4) Clusters identified in the previous step are analyzed, as to whether they comply with the predefined quality requirements (see above), e.g.

- a *learning threshold* of two events, and
- a *confidence threshold* of 0.5.

(5) Facts and related attributes that represent the clusters identified in the previous step are generated and stored in a fact repository. The following fact plus attributes can be generated from the events and the learning parameters:

UserId=12, Price=Medium, Trendiness=Low (*support*=3, *confidence*=0.6).

The cluster described by this fact contains three events and the relative size of the cluster as an indicator for confidence is three out of five events, hence 0.6.
(6) The application can employ facts learned by Learn Sesame for adaptation purposes.

As can be seen from Figure 6, applications and Learn Sesame communicate in an asynchronous manner: events are buffered in the Event Queue for further processing by Learn Sesame and facts are buffered in the Fact Repository for further processing by applications. This allows both applications and Learn Sesame to carry out most of the processing concurrently. In practice, most processes can even be executed

in parallel since applications and Learn Sesame normally run on different computers. Likewise, Learn Sesame exploits the inherent concurrency in the cluster identification and cluster analysis processes by delegating concurrent tasks to different program threads wherever possible (see the 'concurrent' arrows on the right side of Figure 6). The architecture of Learn Sesame therefore seems to support deployment to real world settings with their inherent demand for performance and scalability in terms of user modeling workload very well[27].

A feature that sets Learn Sesame apart from other commercial user modeling systems is its support for domain modeling by means of a model definition language (MDL). In Learn Sesame, a domain model comprises objects and events. Objects represent domain entities (e.g., the user who is assigned the id number 12, a web document describing a particular product). Object type definitions represent categories of domain entities and their possible object attributes (e.g., attributes of a web document like name, creator, modifier, size, date, keywords). Events typically affect one or more objects and are described in terms of changes to objects (e.g., buying a specific product, requesting a web document describing a particular product). Event types define categories of related events (e.g., buying products, requesting web documents).

The domain model is the indispensable basis for both the applications and Learn Sesame. Applications rely on Learn Sesame's domain model for appropriately assembling and communicating events. Learn Sesame's domain independent learning algorithms rely on the domain model for seizing relevant information about the domain at hand. Especially the meaning of comparability, similarity, and proximity of events is of paramount importance for the clustering process (in the simple example presented earlier, we implicitly assumed a semantics for these concepts). In fact, comparability is defined by the object types of the domain model (i.e., two objects are comparable if they belong to the same object type). Similarity is defined by a subset of object attributes and associated values (i.e., two objects are similar if they are comparable and if they are identical with respect to a subset of attributes). Proximity is defined by the distance between attribute values, measured for example with domain-specific distance metrics (e.g., the time stamps of two web page requests are regarded as close to one another if they differ by less than a minute, and are assumed to be identical if they differ by less than fifteen seconds)[28]. It seems that despite (or perhaps because) of this expressiveness of MDL, domain modeling has been rigorously restricted in recent versions of Learn Sesame (i.e., versions greater than 1.2) to a fixed model that only comprises a single pre-defined object type (namely 'web document') and a single event type (namely

[27] In practice, scalability is often more important than absolute performance since scalability refers to the system's performance relative to an increasing workload (e.g., in terms of number of users, number of events and facts submitted per second, and the size and complexity of the domain model).

[28] For more information on Learn Sesame's domain modeling and clustering algorithms we refer to Caglayan et al. (1997).

'web document request'). The rationale behind this step was probably to simplify the domain modeling process for web-based applications.

Extensibility, integration of external user and usage information. Due to the MDL restrictions recently introduced by Bowne, Learn Sesame's learning is confined to the identification and analysis of single attributes only. No further technical means (e.g. APIs) are provided for integrating complementary user modeling products and implementing custom extensions. Integration of events from user and usage information that is external to Learn Sesame (e.g., purchase data from legacy systems, demographic data from a client's database) is supported via a bulk loading interface (see Figure 7). As already mentioned, it is however in the responsibility of applications to collect, assemble, and communicate appropriate event vectors from external data. Exporting user and usage data for use with applications other than Learn Sesame (e.g., software for reporting and visualization) is supported via a reporting facility (see Figure 7). This facility also provides access to conventional reports.

Privacy. Open Sesame, and now Bowne, committed themselves to a strong privacy policy which is based on the following principles (Open Sesame, 2000; Bowne, 2000):

- *informed consent*: request permission from a user before acquiring, using, and especially sharing personal data with third parties,
- *anonymity*: a user has the right to remain anonymous (e.g., when visiting a web site),
- *profile termination*: a user has the right to access, control, and erase any personal information stored about her at any time[29],
- *information security*: all personal information about a user has to be maintained within a secure system.

Bowne is a participant in the P3P project of the World Wide Web Consortium (W3C)[30]. The company claims that Learn Sesame will not only comply to the OPS specification, but to the entire P3P standard after its completion. Bowne encourages its customers to adopt a privacy policy that is similar to their own.

Architecture, software and hardware. Figure 7 depicts the architecture of Learn Sesame, which comprises the following components:

- *Ambassador*: integrates with applications (e.g., web server, e-commerce server) by offering various APIs for communicating events and facts at runtime;
- *Knowledge manager*: transfers events and facts between the Ambassador and the Learning Engine, implements the Event Queue and the Fact Repository

[29]For example, Learn Sesame's API offers functionality for deleting selected parts of a user model including the model as a whole.

[30]The World Wide Web Consortium (W3C) develops common web protocols and aims at ensuring their interoperability. Their freely available specifications called 'Recommendations' have a high impact on the further development of the web. W3C has more than 400 member organizations from around the world.

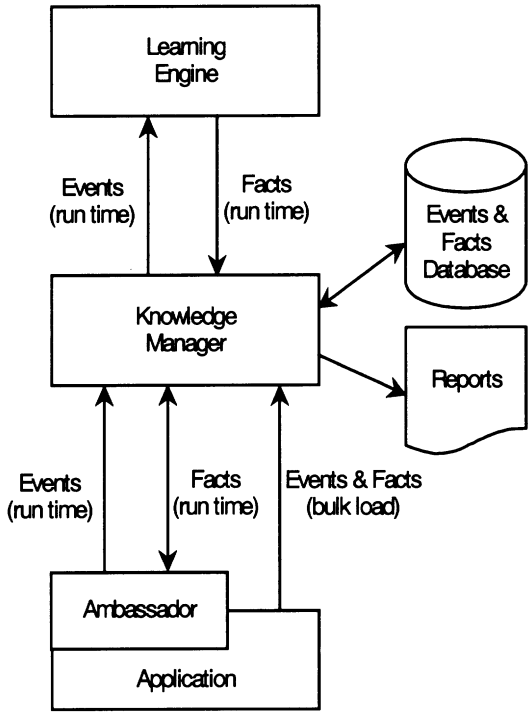

Figure 7. Architecture of Learn Sesame (based on (Open Sesame, 2000)).

(see Figure 6), provides a bulk loading interface for legacy data, and reporting facilities;

- *Learning Engine*: analyzes events for patterns and reports facts describing regularities found.

The communication between these components is carried out via CORBA. The minimum hardware configuration required by Learn Sesame is a single server with a Pentium II processor running at 266 MHz, 64 Mb of RAM, and 50 Mb of free disk space. Bowne recommends a configuration of at least two servers, each of them being better equipped than the one listed above. Supported operating systems are Windows NT and probably SGI Irix and Sun Solaris as well[31]. Connection to a web server or e-commerce server is established via the Ambassador component, which provides local interfaces for Java, C/C++, and Active X. All ODBC-compliant databases including those from Oracle, Sybase, Informix, and Microsoft can be accessed by Learn Sesame.

Client base. Learn Sesame customers are not listed (nor even mentioned) on the web sites of Bowne and Open Sesame, with the exception of Ericsson. We assume

[31]Open Sesame (1998) claims that Learn Sesame supports all three operating systems, whereas Open Sesame (1999) reports that Learn Sesame supports Windows NT only.

that the reason for this is a rather small customer base compared, e.g. to the ones of Net Perceptions and ATG. The only web site that is reported to employ Learn Sesame online is its showcase 'eGenie'.

Similar products. Open Sesame can be considered an early pioneer of personalization, both in research and commercial environments. Despite its early market entry and its sophisticated features, there is, to the best of our knowledge, no commercial system on the market that is comparable to Learn Sesame. In user modeling research, similar work was conducted for example by Orwant (1995) and more recently by Zukerman et al. (1999) in the field of navigation patterns on the World Wide Web. Outside the user modeling community, plenty of related work was carried out in the area of (web) log file mining. Recent research prototypes are described e.g. in Fuller and de Graaf (1996), Spiliopoulou and Faulstich (1999), and Cadez et al. (2000). Examples of commercial products include 'Accrue' (Accrue, 2000) and 'Knowledge Web Miner' from Angoss (2000). The main difference between these systems and Open Sesame! lies in their focal points: Open Sesame! (and later Learn Sesame) were designed for unsupervised and online learning in (near) real time. Applications can control the nature of patterns learned by Open Sesame! as well as the learning process itself. In contrast, many of the related research prototypes and commercial products have been designed for supervised learning of a relatively small (and often fixed) number of patterns that takes place offline. The implications of this difference are manifold, for example regarding data acquisition and representation (see Section 5).

5. Discussion

In Table 1, we summarize the products reviewed so far along the product requirements introduced in Section 3 (for reasons of brevity, we omit the requirements 'client base' and 'similar products'). In order to facilitate orientation, we copy the requirement names as run-in headings in italics.

Functionality and input data. For GroupLens and Personalization Server, input data are restricted to predefined information about the system's usage, and, in the case of Personalization Server, also about the user and the usage environment. Custom extensions can be implemented for Personalization Server on top of the monitoring facilities already provided by Application Server. Compared to the restricted set of input data for GroupLens and the rather tight integration of Personalization Server with a single user-adaptive application (environment), FrontMind's configuration facilities for input data and Learn Sesame's domain modeling facilities with their inherent application independence and flexibility seem to be clearly superior. With Learn Sesame, application programmers can communicate information about the domain at hand and control the associated learning process at an appropriate level of abstraction. It is hardly understandable why these powerful domain modeling facilities have been severely restricted. The provision of a transparent MDL wrapper layer that eases domain model development,

Table 1: Summary of Reviewed User Modeling Servers

	GroupLens	Personalization Server	FrontMind	Learn Sesame
Functionality	user modeling	user modeling, adaptation control	user modeling, adaptation control	user modeling
Data acquisition: Input	predefined usage data, mainly about navigation, ratings, shopping cart operations, and purchases	predefined user, usage, and environmental data (extensible by custom programming)	configurable set of usage, user, and environmental data (extensible by system configuration)	arbitrary usage, user, and environmental data, modeled in MDL[32] and encoded in event vectors
Methods	collaborative filtering	simple production rules, stereotypes	simple production rules, Bayesian networks	hierarchical clustering (single attributes)
Representation	implicit, in cache memory	explicit and relational, in JDBC-compliant databases	group models: implicit in proprietary files; other user-related data: relational, in dedicated database management systems	explicit and relational, in ODBC-compliant databases; it is recommended to access this information via APIs only
Extensibility	no	via Open Targeting Adapters	via custom programmed Business Objects	no
Integration of external user and usage information	bulk load at startup time	via Open Content Adapters and Open Profiling Adapters, dynamically and bi-directional	anytime, via existing or custom Business Objects; dynamically and bi-directional	bulk load at any time
Privacy	company privacy policy, contribution to privacy consortia, OPS compliance	no commitment	vague privacy policy, declared P3P compliance	company privacy policy, contribution to privacy consortia, commitment to OPS and P3P for current and future versions
Architecture	two-tier server	two-tier server (deployable in a server cluster)	multi-tier server based on agents that communicate in XML	three-tier server based on CORBA
Software	Java, C/C++, Perl, COM, and CGI	Java (servlets)	Java (servlets), Active X, and sockets	Java, C/C++, and Active X
Hardware	Windows NT, Sun Solaris; IBM AIX (announced)	Windows NT, Sun Solaris, IBM AIX	Windows NT, Sun Solaris	Windows NT, probably Sun Solaris and SGI Irix

[32]For more information on MDL (Model Definition Language) we refer to Section 4.4.

along with sample configurations for selected deployment scenarios, would have been a more appropriate technical solution to facilitate deployment.

Acquisition methods and representation. With regard to acquisition methods, the reviewed products implement complementary rather than competing approaches. GroupLens uses collaborative filtering, Personalization Server offers (simple) production rules that mainly operate on individual user profiles and stereotypes, FrontMind employs (simple) production rules that take advantage of Bayesian networks, and Learn Sesame employs hierarchical clustering. In the following, we briefly discuss these different acquisition methods along a few key requirements for deployment to real-world settings:

- *Scope of applicability.* Despite the fact that GroupLens supports commonly required user modeling tasks in commercial settings (e.g., predicting user interests and preferences from selected data about system usage), its scope of applicability is rather limited. The acquisition methods used by the other products cover a broader range of user modeling tasks (e.g., Learn Sesame can also learn recurrent patterns of observations in usage data) and can take advantage of a broader range of input data types (namely also data about the user and the usage environment).
- *Facility for incremental learning.* Business practices can often be implemented straightforwardly in rule-driven personalization environments. Moreover, rule-driven personalization allows businesses to be very explicit. From a user's point of view, however, the effects of a solely rule-driven personalization are often found to be quite deterministic (Net Perceptions, 2000; Bachem, 1999). Unlike non-deterministic recommendations, rule-driven personalization leaves barely any room for users' serendipity[33]. This is mainly due to the fact that the underlying representation system for user information can hardly deal with uncertainty and with changes in user behavior. Keeping track of changing user interests and preferences in real time is, however, a main motivation for user modeling from a marketing point of view (see Section 1). Even worse, rule design, update and management is primarily a manual process and therefore cumbersome and error-prone.

  Considering (i) the enormous size of real-world web sites; (ii) the heterogeneous personalization requirements of a large number of different users and user groups; (iii) the necessity to regularly update both content and personalization behavior; and (iv) the paramount importance of consistency for the overall usability of applications, user modeling servers like Personalization Server that solely rely on rule-based personalization and

[33]With serendipity, we refer to a well-known phenomenon in hypermedia where users, stimulated by unexpected interesting information provided by the system, abandon or sidetrack from the original search goal (Conklin, 1987; Nielsen, 1990). Similar effects are reported for non-deterministic recommendations provided e.g. by collaborative filtering systems, as opposed to more deterministic recommendations provided e.g. by rule-based systems (Net Perceptions, 2000; Bachem, 1999).

stereotypes seem to have severe shortcomings. Systems like FrontMind that exhibit both deterministic and non-deterministic personalization behavior seem to have a significant competitive advantage.

- *Explicitly represented knowledge.* In GroupLens, user-related information including dynamically calculated groups of users with similar interests and preferences is represented implicitly in cache memory. Applications can access this information only via pre-defined APIs (see Section 4.1). Personalization Server and FrontMind maintain their user profiles explicitly in profile databases. If necessary, applications can access these databases directly, which usually allows for more general exploitation. In addition to user profiles, FrontMind maintains its models of users' behavior in proprietary files. Likewise, Learn Sesame represents a priori available and dynamically acquired user-related information including related evidences (e.g., support, confidence) in a proprietary database. It is highly recommended to access this information via associated APIs only.
- *Employing domain knowledge in the learning process.* As already pointed out earlier (see Section 4.4), Learn Sesame is unique in employing domain knowledge for guiding the learning process, which significantly contributes to the efficiency and scalability of the overall user modeling process.

Extensibility. GroupLens and Learn Sesame do not allow for custom extensions to their acquisition methods, nor do they provide interfaces for user modeling products that offer such kind of functionality. In contrast, Personalization Server allows for such an integration via its pre-defined Open Targeting Adapters (e.g., for GroupLens). Similarly, FrontMind allows for the incorporation of complementary user modeling functionality by custom programming Business Objects. We believe that Manna will make up for this deficiency and offer a dedicated set of Business Objects for integrating complementary user modeling products (e.g., for GroupLens and LikeMinds) in the near future. On ATG's web site, Accrue (2000) and Macromedia are mentioned as additional technology partners. Both companies offer supplemental products for web site traffic analysis and reporting (e.g., for investigating the effectiveness of personalization features). We believe that ATG showcases that a focused product which offers open interfaces for complementary tools can successfully cope with the broad scope of requirements for a personalization platform (which range from acquiring user and usage data to user modeling, the provision of user-related adaptations, user model inspection and analysis, and reporting).

Integration of external user and usage information. Leveraging the assets of user-related information that is dispersed across the enterprise (e.g., client profiles, past purchase data, user segmentations from database marketing) seems to be of paramount importance for businesses (Hagen et al., 1999). The products reviewed support this only to a very limited extent. GroupLens and Open Sesame provide mainly bulk loading facilities for legacy data, whereas Personalization Server

and FrontMind can directly integrate user-related information from external sources and vice versa at runtime (in the case of Personalization Server, this is only of limited value since this information cannot be incorporated in personalization rules within the Personalization Control Center). In general, central user modeling servers that allow for the integration of existing information sources about users and, vice versa, enable direct access to information stored in user models, can provide the platform for more holistic personalization (see Section 2). In order to achieve this, all relevant user-related information including meta-data has to be (virtually) fused into a single source, regardless of physical details like representation, management system, location, operating system, and network protocol (Truog et al., 1999; Fink, 2001). LDAP[34]-compliant Directory Servers and associated meta-directory facilities (e.g., from Netscape (2000) and iPlanet (2000)) provide such functionality and therefore seem to offer a promising platform for implementing user modeling servers. On top of this platform, dedicated user modeling components can be implemented that communicate with the server platform via CORBA (for the design and implementation of such a system we refer to Fink (2000)). From a client's and application programmer's point of view, this allows for a transparent access to user-related information and associated user modeling services with commonly available applications (e.g., WWW browsers, e-mail clients) and tools (e.g., directory browsers) via a standardized protocol (e.g. LDAP).

Privacy. ATG, and to a lesser extent also Manna, seem to be rather careless regarding privacy, compared for example to the efforts undertaken by Net Perceptions and Bowne. This is somewhat surprising since many tool vendors, their customers and the (online) marketing industry actively propagate and contribute to self-regulation with regard to privacy, in order to prevent governments from passing probably more restrictive privacy laws. The overall aim of self-regulation is to increase customer confidence. Means to this end include (i) privacy initiatives (in the U.S. e.g. from the Federal Trade Commission (FTC, 2000) and the Direct Marketing Association (DMA, 2000)), (ii) practices and policies (e.g., Bowne (2000), Net Perceptions (2000), Amazon (2000), TRUSTe (2000), Electronic Privacy Information Center (EPIC, 2000)), and (iii) technologies and standards (e.g., cookies[35] (Kristol and Montulli, 1997), P3P (W3C, 2000), OPS (W3C, 2000)). For an overview on security and privacy issues in user modeling we refer to Schreck (2000).

Architecture. In order to achieve flexibility and scalability, the user modeling servers reviewed in this article take different architectural approaches. GroupLens and Personalization Server both rely on a two-tier architecture (namely a database tier and a user modeling tier), Learn Sesame relies on an additional third tier (for

[34]LDAP (Lightweight Directory Access Protocol (Howes and Smith, 1997; Howes et al., 1999; Wilcox, 1999)) is a standardized application programmer interface for accessing information about relevant characteristics of users, devices and services on a network.

[35]Cookies are short pieces of textual information (e.g., a unique ID, a user password) that are used for identifying a computer (i.e., not necessarily a specific user) each time a web site is visited. Cookies are sent by web servers and locally stored by browsers.

the Knowledge Manager), and FrontMind is based on a multi-tier architecture. Although Learn Sesame takes advantage of an additional third tier, its granularity of distribution and the associated scalability hardly surpasses that of GroupLens and Personalization Server. The main reason is that in all three systems, the 'critical' user modeling functionality is incorporated into a single tier and hence cannot be distributed over a network of computers. Likewise, the replicability of user modeling components is currently not supported and not planned to be available in the future. An example that motivates this requirement are different learning strategies that can be accomplished by using differently configured versions of the same learning engine: assumptions about users' interests and preferences can be acquired through rather quick and volatile learning in one Learning Engine, whereas assumptions about users' knowledge that probably require a slower and more solid learning process can be acquired in a second Learning Engine.

A user modeling server thus should allow for (i) the flexible distribution of components across a network of computers, according to resource and availability requirements, (ii) the employment of differently configured instances of a user modeling component, and (iii) the integration of complementary user modeling server products. Due to its sophisticated agent-based architecture, FrontMind should be clearly superior regarding flexibility and scalability (the XML-based communication between agents may not be very efficient though). However, and to the best of our knowledge, FrontMind does not provide replication facilities and Business Objects for integrating complementary user modeling products at the moment.

Software and hardware. The APIs supported by the reviewed user modeling servers reflect the targeted kind of user-adaptive applications. Personalization Server provides interface support for Java only (i.e., the programming language used in ATG's products) and, via Application Server, for dedicated web servers (namely 'Apache HTTPD', Microsoft's 'Internet Information Server', and Netscape's 'Enterprise Server' and 'FastTrack Server'). Group Lens, FrontMind, and Learn Sesame provide a variety of interfaces for several programming languages (e.g., Java, C/C++, Perl). Moreover, they support common component frameworks, e.g. Active X, COM, CGI-compliant web servers, and one or more e-commerce servers (e.g., Broadvision's One-To-One, Vignette's StoryServer (Vignette, 2000), Microsoft's Site Server, Allaire's 'ColdFusion' (Allaire, 2000), and IBM's 'Net.Commerce' (IBM, 2000)).

6. Summary and Conclusion

This article surveyed and compared selected commercial user modeling servers against the background of requirements in real-world environments (mostly web-based e-commerce). In order to support this analysis, we defined a requirements catalogue and categorized the reviewed products.

We found that the reviewed commercial user modeling servers offer in part sophisticated deployment-supporting features, e.g. regarding the offered acquisition

methods and the supported platforms (although the degree of sophistication varies from product to product). Given the potential of centralized user modeling, we believe that there is considerable room for future improvements of these products, especially regarding (i) acquisition methods, with a focus on the integration of domain knowledge and method mix at deployment time, (ii) extensibility, and (iii) integration of user-related information that is external to the user modeling server. Work on these issues is already underway (see related references in Section 5).

Acknowledgements

This work was supported by a Grant of the European Media Lab (EML), Heidelberg, Germany. The authors wish to thank the four anonymous UMUAI reviewers and Judy Kay for their valuable comments on previous versions of this article.

References

1to1web: 2000, One-to-One Web Marketing Online. *http://www.1to1web.com.*

Åberg, J. and Shahmehri, N.: 1999, Web assistants: Towards an intelligent and personal web shop. *Proceedings of Second Workshop on Adaptive Systems and User Modeling on the World Wide Web at WWW-8*, Toronto, Canada, and UM-99, Banff, Canada, pp. 5–12. *http://www.contrib.andrew.cmu.edu/~plb/WWWUM99_workshop/aberg/aberg.html*

Abrams, C., Bernstein, M., deSisto, R., Drobik, A. and Herschel, G.: 1999, E-Business: The Business Tsunami. *Proceedings of Gartner Group Symposium/ITxpo*, Cannes, France.

Accrue: 2000, Accrue Software. *http://www.accrue.com*

Allaire: 2000, ColdFusion. Allaire. *http://www.allaire.com/Products/ColdFusion*

Allen, C., Kania, D. and Yaeckel, B.: 1998, *Internet World Guide to One-To-One Web Marketing*. New York (NY): John Wiley and Sons.

Alspector, J., Kolcz, A. and 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.

Amazon: 2000, Your Privacy. Amazon.com. *http://www.amazon.com/exec/obidos/subst/misc/policy/privacy.html*

Ambrosini, L., Cirillo, V. and Micarelli, A.: 1997, A hybrid architecture for user-adapted information filtering on the world-wide web. In: A. Jameson, C. Paris and C. Tasso (eds.), *User Modeling: Proceedings of the Sixth International Conference*. Wien, New York (NY): Springer. 59–61. *http://www.um.org/um_97/gz/AmbrosiniL.ps.gz*

Angoss: 2000, Angoss Software. *http://www.angoss.com*

Appian: 2000a, Appian. *http://www.appiancorp.com*

Appian: 2000b, Appian Web Personalization Report. Appian. *http://www.appiancorp.com/awpr.asp*

Ardissono, L. and Goy, A.: 1999, Tailoring the interaction with users in electronic shops. In: J. Kay (ed.), *UM99 User Modeling: Proceedings of the Seventh International Conference*. Wien, New York (NY): Springer, 35–44. *http://www.um.org/um_99/Proc/ardissono.pdf*

Ardissono, L., Goy, A., Meo, R. and Petrone, G.: 1999, A configurable system for the construction of adaptive virtual stores. *World Wide Web* **2**(3), 143–159.

Ardissono, L. and Goy, A.: 2000, Tailoring the Interaction with Users in Web Stores. Submitted.

ATG: 2000, Dynamo Product Suite. Art Technology Group. *http://www.atg.com/products/highlights*

Autonomy: 2000, Autonomy Systems. *http://www.autonomy.com*

Bachem, C.: 1999, Profilgestütztes Online Marketing. In: S. Tjoa (ed.), *Personalisierung im E-Commerce.* Hamburg, Germany, Section 13.

Barnea, G.: 1999, Intelligent Agent Communities. Manna. *http://www.mannainc.com/downloads/intlagnt.zip*

Bertram, F.: 2000, *VerBose – Verteilte Architektur für Benutzermodellierungssysteme*. Master Thesis, Dept. of Computer Science, University of Koblenz-Landau, Germany.

Billsus, D. and Pazzani, M.: 2000, User modeling for adaptive news access. **10**, 147–180 (this issue).

Blaze: 2000, Blaze Software. *http://www.blazesoftware.com*

Blue Martini: 2000, Blue Martini Software. *http://www.bluemartini.com*

Bowne: 2000, Bowne & Co. *http://www.bowne.com*

Brajnik, G. and 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. and Kadie, C.: 1998, Empirical analysis of predictive algorithms for collaborative filtering. *Proceedings of the Fourteenth Annual Conference on Uncertainty in Artificial Intelligence* (UAI-98). San Francisco (CA): Morgan Kaufmann, pp. 43–52.

BroadVision: 2000, BroadVision. *http://www.broadvision.com*

Brusilovsky, P., Kobsa, A. and Vassileva, J. (eds.): 1998, *Adaptive Hypertext and Hypermedia.* Dordrecht: Kluwer Academic Publishers.

BusinessWeek: 1999, Global 1000. BusinessWeek online. *http://www.businessweek.com/1999/99_28/g1000.htm*

Cadez, I., Heckerman, D., Meek, C., Smyth, P. and White, S.: 2000, Visualization of navigation patterns on a Web site using model-based clustering. *Proceedings of the Sixth ACM Conference on Knowledge Discovery and Data Mining*. Boston (MA), forthcoming.

Caglayan, A. and Snorrason, M.: 1993, On the relationship between generalized equality clustering and ART 2 neural networks. *World Congress on Neural Networks.* Portland (OR), July 1993.

Caglayan, A., Snorrason, M., Jacoby, J., Mazzu, J., Jones, R. and Kumar, K.: 1997, Learn Sesame – a Learning Agent Engine. *Applied Artificial Intelligence* **11**, 393–412.

Conklin, J.: 1987, Hypertext: An Introduction and Survey. *IEEE Computer* September 1987, pp. 17–41.

Cooperstein, D., Delhagen, K., Aber, A. and Levin, K.: 1999, *Making Net Shoppers Loyal*. Cambridge (MA): Forrester Research.

DMA: 2000, Direct Marketing Association. *http://www.the-dma.org*

Eklund, J. and Brusilovsky, P.: 1998, The value of adaptivity in hypermedia learning environments: a short review of empirical evidence. In: P. Brusilovsky and P. De Bra (eds.), *Proceedings of Second Adaptive Hypertext and Hypermedia Workshop at the Ninth ACM International Hypertext Conference Hypertext'98.* Pittsburgh (PA), pp. 11–17. *http://wwwis.win.tue.nl/ah98/Eklund.html*

EPIC: 2000, Electronic Privacy Information Center. *http://www.epic.org*

Finin, T. W.: 1989, GUMS: A general user modeling shell. In: A. Kobsa and W. Wahlster, (eds.), *User Models in Dialog Systems.* Berlin, Heidelberg: Springer. 411–430.

Fink, J., Kobsa, A., and 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.

Fink, J.: 2001, *User Modeling Servers – Requirements, Design, and Implementation.* Doctoral Dissertation, Dept. of Mathematics and Computer Science, University of Essen, Germany (forthcoming).

Forrester: 2000, Forrester Research. *http://www.forrester.com*

FTC: 2000, Federal Trade Commission. *http://www.ftc.gov*

Fuller, R. and de Graaf, J.: 1996, Measuring User Motivation from Server Log Files. In: *Proceedings of the Microsoft Conference Designing for the Web – Empirical Studies*, Redmond (WA).
http://www.microsoft.com/usability/webconf/fuller/fuller.htm

Good, N., Schafer, J., Konstan, J., Borchers, A., Sarwar, B., Herlocker, J. and Riedl, J.: 1999, Combining collaborative filtering with personal agents for better recommendations. In: *Proceedings of the 1999 Conference of the American Association of Artificial Intelligence (AAAI-99)*, 439–446. *http://www.cs.umn.edu/Research/GroupLens/aaai-99.pdf*

Goscinski, A.: 1991, *Distributed Operating Systems: The Logical Design.* Sydney: Addison-Wesley.

Gustos: 2000, Gustos Software. *http://www.gustos.com*

Hagen, P., Manning, H. and Souza, R.: 1999, *Smart Personalization.* Cambridge (MA): Forrester Research.

Herlocker, J., Konstan, J., Borchers, A. and Riedl, J.: 1999, An algorithmic framework for performing collaborative filtering. In: M. Hearst, F. Gey and R. Tong (eds.), *Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval.* New York (NY): ACM. 230–237.
http://www.cs.umn.edu/Research/GroupLens/algs.pdf

HNC: 2000 Software. *http://www.hnc.com*

Hof, R., Green, H. and Himmelstein, L.: 1998, Now it's YOUR WEB. *Business Week*, October 5: 68–74.

Howes, T. and Smith, M.: 1997, *Ldap: Programming Directory-Enabled Applications with Lightweight Directory Access Protocol*. Indianapolis (IN): Macmillan.

Howes, T., Smith, M. and Good, G.: 1999, *Understanding and deploying LDAP directory services*. London: Macmillan.

IBM: 2000, IBM Net.Commerce. IBM.
http://www-4.ibm.com/software/webservers/commerce/servers

ICONOCAST: 1999, More concentrated than the leading brand. ICONOCAST.
http://www.iconocast.com/icono-archive/icono.102199.html

iPlanet: 2000, iPlanet Directory and Security Services. Sun-Netscape Alliance.
http://www.iplanet.com/products/infrastructure/dir_security/index.html

Jörding: 2000, Short-term user modeling for adaptive product presentations on the World Wide Web. Submitted.

Jupiter: 2000, Jupiter Communications. *http://www.jup.com*

Kay, J.: 1995, The um toolkit for reusable, long term user models. *User Modeling and User-Adapted Interaction* **4**(3), 149–196.

Kobsa, A., Müller, D. and Nill, A.: 1994, KN-AHS: An adaptive hypertext client of the user modeling system BGP-MS. In: *Proceedings of the Fourth International Conference on User Modeling.* Hyannis (MA), pp. 99–105. Reprinted in M. Maybury and W. Wahlster (eds.): 1998, *Intelligent User Interfaces.* San Mateo (CA): Morgan Kaufman. pp. 372–378.
http://ics.uci.edu/~kobsa/papers/1994-UM94-kobsa.ps

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

Kobsa, A.: 2001, Generic user modeling systems. *User Modeling and User-Adapted Interaction* **11**(1), 10 Year Anniversary Issue.

Kobsa, A., Koenemann, J. and Pohl, W.: 2001, Personalized hypermedia presentation techniques for improving customer relationships. To appear in *The Knowledge Engineering Review.*
http://ics.uci.edu/~kobsa/papers/Kobsa-PHPT-draft.pdf

Konstan, J., Miller, B., Maltz, D., Herlocker, J., Gordon, L. and Riedl, J.: 1997, GroupLens: Applying collaborative filtering to Usenet News. *Communications of the ACM* **40**(3), 77–87.
http://www.acm.org/pubs/citations/journals/cacm/1997-40-3/p77-konstan

Kristol, D. and Montulli, L.: 1997, HTTP State Management Mechanism.
ftp://ftp.isi.edu/in-notes/rfc2109.txt

Machado, I., Martins, A. and Paiva, A.: 1999, One for all and all in one: a learner modelling server in a multi-agent platform. In: J. Kay (ed.), UM99 User Modeling: *Proceedings of the Seventh International Conference.* Wien, New York (NY): Springer, pp. 211–221.
http://www.um.org/um_99/Proc/paiva.pdf

Macromedia: 2000, LikeMinds. Macromedia.
http://ebusiness.macromedia.com/products/likeminds/

Manna: 1999a, FrontMind for Marketing. Manna.
http://www.mannainc.com/downloads/fmwhite.zip

Manna: 1999b, Automated Distributed Intelligence. Manna.
http://www.mannainc.com/downloads/adiwhite.zip

Manna: 2000a, Manna. *http://www.mannainc.com*

Manna: 2000b, FrontMind. Manna. *http://www.mannainc.com/downloads/whitepaper.zip*

Manna: 2000c, Online Personalization for E-commerce: The Manna Advantage. Manna.
http://www.mannainc.com/downloads/personalization.zip

Manna: 2000d, FrontMind Components. Manna.
http://www.mannainc.com/products_stage_technical.html

Marketing 1to1: 2000, Marketing 1to1 – Marketplace1to1. Peppers and Rogers Group.
http://search.marketplace1to1.com

Microsoft: 2000, Product and Technology Catalog. Microsoft.
http://www.microsoft.com/products

Moore, J. and Paris, C.: 1992, Exploiting user feedback to compensate for the unreliability of user models. *User Modeling and User-Adapted Interaction* **2**(4), 331–365.

Net Perceptions: 2000, Net Perceptions. *http://www.netperceptions.com*

Netscape: 2000, Directory and LDAP Developer Central. Netscape.
http://developer.netscape.com/tech/directory/index.html

Nielsen, J.: 1990, The Art of Navigating through Hypertext. *Communications of the ACM* **33**(3): 296–310.

Nielsen: 2000, Nielsen Media Research. *http://www.nielsenmedia.com*

Open Sesame 2000, Open Sesame. Bowne & Co. *http://www.opensesame.com*

Orfali, R., Harkey, D. and Edwards, J.: 1994, *Essential Client/Server Survival Guide.* New York (NY) and Singapore: Wiley and Sons.

Orwant, J.: 1995, Heterogeneous learning in the Doppelgänger user modeling system. *User Modeling and User-Adapted Interaction* **4**(2), 107–130.

Paliouras, G., Karkaletsis, V., Papatheodorou, C. and Spyropoulos, C.: 1999, Exploiting learning techniques for the acquisition of user stereotypes and communities. In: J. Kay (ed.), UM99 User Modeling: *Proceedings of the Seventh International Conference.* Wien, New York (NY): Springer, pp. 169–178.
http://www.um.org/um_99/Proc/karkaletsis.pdf

Peppers, D. and Rogers, M.: 1993, *The One to One Future: Building Relationships One Customer at a Time.* New York (NY): Currency Doubleday.

Peppers, D. and Rogers, M.: 1997, *Enterprise One to One: Tools for Competing in the Interactive Age*. New York (NY): Currency Doubleday.

Peppers, D., Rogers, M. and Dorf, B.: 1999, *The One to One Fieldbook*. New York (NY): Currency Doubleday.

Personalization: 2000, personalization.com. *http://www.personalization.com*

Pohl, W. and Nick, A.: 1999, Machine learning and knowledge-based user modeling in the LaboUr approach. In: J. Kay (ed.), *UM99 User Modeling: Proceedings of the Seventh International Conference*. Wien, New York (NY): Springer, 179–188. *http://www.um.org/um_99/Proc/pohl.pdf*

Popp, H. and Lödel, D.: 1996, Fuzzy techniques and user modeling in sales assistants. *User Modeling and User-Adapted Interaction* **5**(3–4), 349–370.

Reagle, J. and Cranor, L.: 1999, The platform for privacy preferences. *Communications of the ACM* **42**(2), 48–55.

Reichheld, F.: 1996, *The Loyalty Effect*. Boston (MA): Harvard Business School Press.

Resnick, P., Iacovou, N., Sushak, M., Bergstrom, P. and Riedl, J.: 1994, GroupLens: An open architecture for collaborative filtering of Netnews. In: *Proceedings of the Conference on Computer Supported Cooperative Work*. pp. 175–186. *http://www.acm.org/pubs/citations/proceedings/cscw/192844/p175-resnick/*

Rich, E.: 1979, User modeling via stereotypes. *Cognitive Science* **3**, 329–354.

Rich, E.: 1983, Users are individuals: Individualizing user models. *Journal of Man-Machine Studies* **18**, 199–214.

Rich, E.: 1989, Stereotypes and user modeling. In: A. Kobsa and W. Wahlster (eds.), *User Models in Dialog Systems*. Berlin: Springer Verlag, pp. 35–51.

RightPoint: 2000, RightPoint Software. *http://www.rightpoint.com*

Schafer, J., Konstan, J. and Riedl, J.: 1999, Recommender systems in electronic commerce. *Proceedings of the ACM Conference on Electronic Commerce (EC-99)*. Denver (CO): ACM. pp. 158–166. *http://www.cs.umn.edu/Research/GroupLens/ec-99.pdf*

Schreck, J.: 2000, *Security Issues in User Modeling*. Doctoral Dissertation, Dept. of Mathematics and Computer Science, University of Essen, Germany (forthcoming).

Singh, J.: 2000, Privacy, Information, and Consumer Value. ART Technology Group. *http://www.atg.com/news/white-papers/privacy.html*

Snorrason, M. and Caglayan, A.: 1994, Generalized ART2 algorithms. *World Congress on Neural Networks*. San Diego (CA), June 1994.

Specht, M.: 1998, Empirical evaluation of adaptive annotation in hypermedia. *Proceedings of the ED-MEDIA98*, Freiburg, Germany, pp. 1327–1332.

Specht, M. and Kobsa, A.: 1999, Interaction of domain expertise and interface design in adaptive educational hypermedia. *Proceedings of the Second Workshop on Adaptive Systems and User Modeling on the World Wide Web at WWW-8*, Toronto, Canada, and UM-99, Banff, Canada, pp. 89–93. *http://ics.uci.edu/~kobsa/papers/1999-WWW8UM99-kobsa.pdf*

Spiliopoulou, M. and Faulstich, L.: 1999, A tool for web utilization analysis. In: *Proceedings of the International Workshop on the Web and Databases - WebDB'98*. Berlin, Heidelberg: Springer, 184–203. *http://www.dia.uniroma3.it/webdb98/papers/12.ps*

SUN: 2000, Java Technology Index. SUN Microsystems. *http://www.sun.com/products-n-solutions/software/api/java.html*

Tanenbaum, A.: 1992, *Modern Operating Systems*. Englewood Cliffs (NJ): Prentice Hall.

Truog, D., Bernoff, J., Ritter, T. and Goldman, H.: 1999, *Centralize Access Control Now*. Cambridge (MA): Forrester Research.

TRUSTe: 2000, TRUSTe. *http://www.truste.org*

Vignette: 2000, Vignette. *http://www.vignette.com*

W3C: 2000, Platform for Privacy Preferences (P3P) Project. W3C. *http://www.w3.org/P3P*
Weber, G. and Specht, M.: 1997, User modeling and adaptive navigation support in WWW-based tutoring systems. In: A. Jameson, C. Paris and C. Tasso (eds.), *User Modeling: Proceedings of the Sixth International Conference.* Wien, New York (NY): Springer. pp. 289–300. *http://www.um.org/um_97/gz/WeberG.ps.gz*
Yimam, D. and Kobsa, A.: 2001, Expert finding systems for organizations: Problem and domain analysis and the DEMOIR approach. In M. Ackerman, A. Cohen, V. Pipek and V. Wulf, (eds.), *Beyond Knowledge Management: Sharing Expertise*. forthcoming
Zukerman, I., Albrecht, D. and Nicholson, A.: 1999, Predicting users' requests on the WWW. In: J. Kay (ed.), *UM99 User Modeling: Proceedings of the Seventh International Conference.* Wien, New York (NY): Springer, pp. 275–284. *http://www.um.org/um_99/Proc/zukerman.pdf*

Authors' vitae

Josef Fink is a co-founder and CTO of humanIT Human Information Technologies GmbH, St. Augustin, Germany. He received a Diploma in Management Information Systems from Furtwangen University of Applied Sciences and a Master's Degree in Information Science from the University of Konstanz. His primary interests lie in the areas of server systems for user modeling and user-adaptive systems. Currently he is working for his Ph.D. thesis on user modeling servers.

Alfred Kobsa (*http://www.ics.uci.edu/~kobsa*) is an Associate Professor at the Department of Information and Computer Science of the University of California, Irvine, CA and a Professor of Computer Science at the University of Essen, Germany. Before he was a Director of the Institute FIT at GMD – German National Research Center for Information Technology, and an Associate Professor of Information Systems at the University of Konstanz, Germany. He received his Ph.D. in Computer Science from the Technical University of Vienna. Dr Kobsa's research focuses on user-adaptive information environments, user modeling, information visualization, expert finders, multimedia educational software, and user interfaces for handicapped and elderly people. He is an editorial board member of *World-Wide Web* and *Universal Access in the Information Society*, and was the founding president of *User Modeling, Inc.*