Generic User Modeling Systems [1]

Alfred Kobsa

Donald Bren School of Information and Computer Sciences
University of California, Irvine
Irvine, CA 92697-3440, U.S.A.
kobsa@uci.edu
http://www.ics.uci.edu/~kobsa

Abstract. This chapter reviews research results in the field of Generic User Modeling Systems. It describes the purposes of such systems, their services within user-adaptive systems, and the different design requirements for research prototypes and commercial deployments. It discusses the architectures that have been explored so far, namely shell systems that form part of the application, central server systems that communicate with several applications, and possible future agent-based user modeling systems. Major implemented research prototypes and commercial systems are briefly described.

1   User Modeling Shell Systems

1.1 Historical Development

User modeling is usually traced back to the works of Allen, Cohen and Perrault (see, e.g., [1, 17, 76]) and Elaine Rich [81, 82]. Inspired by their seminal research, numerous application systems in various application areas were subsequently developed that collected different information about, and exhibited different kinds of adaptation to, their current users. Several publications from this time [66, 70, 97] offer comprehensive reviews of first-generation user-adaptive applications.

In this early work, all user modeling was performed by the application system. In most cases, there was no clear distinction between system components that served user modeling purposes and components that performed other tasks. From the mid-eighties onwards, such a separation was increasingly made (e.g., in [2, 38, 46, 88]), but no efforts are reported to make the user modeling component reusable for the development of other user-adaptive systems.

In 1986, Tim Finin published his "General User Modeling System" GUMS [21, 22]. This software allows programmers of user-adaptive applications the definition of simple stereotype hierarchies (see Chapter MT2 [27]). For each stereotype, one can define the Prolog facts describing stereotype members and the rules prescribing the

---

system's reasoning about them. At runtime, GUMS accepts and stores new facts about the user which are provided by the application system, verifies the consistency of a new fact with currently held assumptions, informs the application about recognized inconsistencies, and answers queries of the application concerning the currently held assumptions about the user.

Albeit GUMS was never used together with an application system, it set the stage for future generic (i.e., application-independent) user modeling systems. At the same time, GUMS also defined their basic functionality, namely the provisioning of selected user modeling services at runtime that can be configured during development time. When filled by the developer with application-specific user modeling knowledge at the time of development, these systems would serve as a separate user modeling component in an application system at runtime. Early systems usually included a representation system for expressing the contents of the user model (such as some logic formalism, rules, or simple attribute-value pairs) and a reasoning mechanisms for deriving assumptions about the user from existing ones and for detecting inconsistencies in the user model.

Kobsa [48] seems to be the first author who used the term "user modeling shell system" for such kinds of software tools. The term "shell system", or "shell" for short, was thereby borrowed from the field of Expert Systems. There, van Melle and Buchanan [94] as well as Shortliffe [13] had condensed the experiences made with the medical expert system MYCIN [87] into EMYCIN ("Essential" MYCIN), an "empty" expert system that had to be filled with domain-specific rules for deployment as a "real" expert system. Commercial expert system shells like Knowledge Craft [45], KEE [36] and ART [16] became very popular in the late seventies and early eighties. User modeling "shells" had similar purposes as expert system shells, but the general underlying aims, namely software decomposition and abstraction to support modifiability and reusability, is of course much older than expert system shells.

1.2 Example Systems

Examples of user modeling shell systems after GUMS include the following ones:[2]

UMT. UMT [6] allows the user model developer the definition of hierarchically ordered user stereotypes, and of rules for user model inferences and contradiction detection. Information about the user that is received from the application is classified as invariable premises or (later still retractable) assumptions. When new information is received, stereotypes may become activated and their contents (which describe the respective user subgroups) added to the user model. UMT then applies inference rules (including contradiction detection rules) to the set of premises and assumptions, and records the inferential dependencies. After the firing of all applicable inference rules and the activation of all applicable stereotypes, contradictions between assumptions are sought and various resolution strategies applied ("truth maintenance").

---

[2] Also see [5, 35, 59, 65] for additional systems.

PROTUM. PROTUM [96] represents user model content as a list of constants, each with associated type (i.e., observed, derived from stereotype, default) and confidence factor. It is related to UMT except that it possesses more sophisticated stereotype retraction mechanisms than UMT.

TAGUS. TAGUS [73] represents assumptions about the user in first-order formulas, with meta-operators expressing the different assumption types (such as 'belief', 'goal', 'capability of reasoning'). The system allows for the definition of a stereotype hierarchy and contains an inference mechanism, a truth maintenance system (with different strengths of endorsements for assumptions about the user), and a diagnostic subsystem including a library of misconceptions. It also supports powerful update and evaluation requests by the application, including a simulation of the user (i.e., forward-directed inferences on the basis of the user model) and the diagnosis of unexpected user behavior.

um. um [40, 41] is a toolkit[3] for user modeling that represents assumptions about the user's knowledge, beliefs, preferences, and other user characteristics in attribute-value pairs. Each piece of information is accompanied by a list of evidence for its truth and its falsehood. The source of each piece of evidence, its type (observation, stereotype activation, rule invocation, user input, told to the user) and a time stamp is also recorded. Explanations for components, sources of evidence, and types of evidence sources may be entered as well. At runtime, competing specialized inference processes (the so-called "resolvers") interpret the available evidence and conclude the value of a component. Applications have to decide which resolvers to employ. Users can inspect and edit their user models [42].

2   User Modeling Servers

2.1 Characteristics

The purpose of user modeling servers, like that of user modeling shells, is to separate user modeling functionality from user-adaptive application systems. In contrast to user modeling shell systems, user modeling servers are not a part of but rather independent from application systems (i.e., they are not functionally integrated into the application but communicate with the application through inter-process communication). User modeling servers may reside on the same platform as the application system and only serve one instance of this application [54]. Much more commonly however, they will be part of a local area network or a wide area network and serve more than one application instance at a time (possibly instances for several hundred thousand simultaneous users in the case of personalized e-commerce

---

[3]   From the point of view of the application system, um was more a library of user modeling functions than an independent user modeling component. It therefore is not a user modeling shell in a strict sense.

websites, cf. Chapter AP3 [28]). They communicate with application systems through protocols that both sides support, such as LDAP, ODBC, remote procedure calls, or plain TCP/IP.

A client-server based architecture provides a number of advantages in comparison to embedded user modeling components that we described in Section 1 (see [25] and [4] for more comprehensive discussions):

– All information about the user is maintained in a repository with clearly defined points of access (usually one single access point).

– User information is at the disposal of more than one application at a time.

– User information acquired by one application can be employed by other applications, and vice versa.

– Information about users is stored in a non-redundant manner.

– The consistency and coherence of information gathered by different applications can be more easily ascertained.

– Information about user groups, either available a priori as stereotypes (e.g., [81-84]) or dynamically calculated as user group models (e.g., [72, 75]), can be maintained with low redundancy.

– Methods and tools for system security, identification, authentication, access control and encryption can be applied for protecting user models in user modeling servers [57, 86].

– Complementary user information that is dispersed across the enterprise (e.g., demographic data from client databases, past purchase data from transactional systems, user segmentations from marketing research) can be integrated more easily with the information in the user model repository.

User modeling servers may be "centralized" (e.g., reside on a single platform only). This facilitates their implementation, but exposes them to the typical downsides of centralization such as the need for a permanent network connection and the jeopardy of a single point of failure and potential bottleneck. Most modern user modeling servers, specifically the commercial ones, are therefore distributed across several platforms to increase their performance and availability (typically through CORBA [79]). Most commercial servers also allow the virtual integration of heterogeneous "outside" resources of user information, sometimes to the point that transparent read and write access to these outside resources is possible dynamically at runtime. Increasingly, user modeling servers also allow for the (partial) replication of user modeling server entries, which mitigates the need for a reliable permanent network connection.

A large number of user modeling servers have been developed over the past 15 years, ranging from academic prototypes to commercial systems. In the next section, we briefly describe examples of major research prototypes, and thereafter the currently available commercial systems.

2.2 Examples of Research Prototypes of User Modeling Servers

BGP-MS. BGP-MS [55, 78] allows assumptions about the user and stereotypical assumptions about user groups to be represented in a first-order predicate logic. A subset of these assumptions is stored in a terminological logic. Different assumption types, such as (nested) beliefs and goals as well as stereotypes, were represented in different partitions that could be hierarchically ordered to exploit inheritance of partition contents (a partition together with all its direct and indirect ancestor partitions thereby establishes a so-called *view* of the full user model). Inferences across different assumption types (i.e. partitions) can be defined in a first-order modal logic. The BGP-MS system can be used as a network server with multi-user and multi-application capabilities.

DOPPELGÄNGER. DOPPELGÄNGER [72] accepts information about the user from hardware and software sensors. Techniques for generalizing and extrapolating data from the sensors (such as beta distributions, linear prediction, Markov models) are put at the disposal of user model developers. Unsupervised clustering (see Section MT3 [69]) is available for collecting individual user models into so-called 'communities' whose information serves the purpose of stereotypes. (In contrast to all other user modeling shell systems, membership in a stereotype is probabilistic rather than definite). The different representations of DOPPELGÄNGER are quite heterogeneous. As is the case for um (see Section 1.2), users can inspect and edit their user models.

CUMULATE. CUMULATE [9] is designed to provide user modeling functionality to a student-adaptive educational system (see Chapters MT1 [67] and AP2 [11]). It collects evidence (events) about student learning from multiple servers that interact with the student. It stores students' activities and infers their learning characteristics, which are the basis for an individual adaptation to them. In this vein, external and internal inference agents process the flow of events and update the values in the inference model of the server. Each inference agent is responsible for maintaining a specific property in the inference model, such as the current motivation level of the student or the student's current level of knowledge for each course topic. Brusilovsky et al. [12] describe the interaction of CUMULATE with an Ontology Server, which stores the ontological structures of the taught domain and provides the platform for the exchange of higher-level information about students' knowledge between different user model servers.

Personis. Personis [43] and a cut-down version of it, PersonisLite [15], have the same representational foundations as their predecessor um that was described in Section 1.2. The components from um form objects in Personis that reside in an object layer over Berkeley DB, a nearly relational database system. The object database also structures user models into hierarchically ordered contexts similar to the partitions of BGP-MS (see Section 1.2). It also holds objects which define the views that include components from all levels of the user model context hierarchy. The authors distinguish two basic operations upon this representation: accretion, which involves

the collection of uninterpreted evidence about the user, and resolution, the interpretation of the current collection of evidence (cf. the resolvers in um).

UMS. UMS [24, 52] is a user modeling server that is based on the Lightweight Directory Access Protocol (LDAP). Its Directory Component allows for the representation of different models, such as user and usage profiles as well as system and service models. "Pluggable" User Modeling Components are internal clients of the Directory Component. They can access these models and perform dedicated user modeling task, such as collaborative filtering, domain-based inferences, etc. The stored models can also be accessed by External Clients, such as user-adaptive applications or tools for user model inspection, visualization and statistical analysis. The use of LDAP allows one to base the storage component of user modeling servers on international standards that have been adopted by industry, to distribute user information across a network and to replicate and loosely synchronize such information (both of which often increases the performance, scalability, availability and reliability of a service), and to realize a "virtually centralized distributed architecture" for user models that is internally distributed but provides a common point of access to clients.

2.3 Examples of Commercial User Modeling Servers

Group Lens. Group Lens [71] originally employed various collaborative filtering algorithms [7, 32] for predicting users' interests, based on explicitly provided users ratings, implicit ratings derived from users' navigation, and transaction histories (e.g., shopping basket operations, purchases). GroupLens stored all user ratings in a database, but kept a correlation matrix of all ratings in cache memory during runtime. This created memory problems and huge performance problems on the largest sites. They were solved temporarily with reduced-size models being selected statistically (with careful sampling, the reduced-size models did not show much quality degradation). Net Perceptions eventually moved to item-item models, which can be truncated substantially without much loss of quality [68].

ATG Adaptive Scenario Engine. ATG Adaptive Scenario Engine [3] allows for the definition of rules that assign individual users to one or more user groups (e.g., customer segments) based on their demographics, their system usage, and their software, hardware and network environments. Rules can also be defined for inferring individual assumptions about the user from his or her navigation behavior, and for personalizing the content of web pages. The operation of Personalization Server thus follows very much the "stereotype approach" from classical user modeling research (see Chapter MT2 [27]). Customer data from legacy databases can be integrated via SQL, XML and Web Services.

enQuire™ Identity Server. enQuire [20] is a multi-functional server product with an embedded virtual directory engine. It supports the development of user modeling

servers by introducing a flexible virtualization layer between multiple repositories and applications that provide user data via LDAP, ODBC or an API. The enQuire Server component stores information about user data sources in an enterprise (as well as the structure of this data), enforces security policies and rule-based access control, federates user data from connected information sources, applies rules to filter and transform user data sets, and presents consolidated user data in a standard format. The results of the federation process can be stored in a persistent cache, which eliminates the dependency on source-specific data structures. enQuire supports the assignment of users to static or dynamically constructed user groups. enQuire plug-ins are customization components that enable developers to describe certain actions to be executed under specific circumstances or at desired points in the request execution process.

Other identity management and user provisioning systems. In addition to the commercial systems mentioned above, a large number of so-called "Identity Management Systems" or "User Provisioning Systems" are commercially available which to some extent include important functionality of a user modeling server (see, e.g., [18, 61, 91, 93]). Such functionality includes one or more of the following: integration of disparate user data into a single centralized repository, federated provisioning of disparate user data, account linking, policy-based access control, user account management, and support for privacy and security audits. These systems however lack other essential functionality, such as inference capabilities on the basis of user data (including the assignment of users to user groups), or triggers for personalization methods. They can therefore not yet be regarded as user modeling servers.

3 Required Services and Characteristics of Generic User Modeling Systems

3.1 From Ingredients to Services of Generic User Modeling Systems

Developers of user modeling shell systems (see Section 1) and early user modeling servers (see Section 2) aimed at condensing basic structures and processes into these systems that they deemed important for user-adaptive application systems (e.g., certain knowledge representation systems, inference mechanisms, truth maintenance systems, and tell/ask interfaces). For identifying such important structures and processes, developers mostly relied on their intuitions and/or their experience through prior work on user-adaptive systems. Efforts to put these decisions on more empirical grounds were seemingly only made by Kleiber [44] and Pohl [77, 78]. Even these authors however merely identified individual user-adaptive application systems in the literature that would have profited from the functionality of their own shell system, rather than conducting a comprehensive review of prior user-adaptive systems, and determining current and predicting future needs of user-adaptive application systems.

In an attempt to extend the de facto definition of user modeling shells introduced by GUMS and to avoid characterizing user modeling shell systems via internal structures and processes, Kobsa [49] listed the following frequently found *services* of such systems:

– "the representation of assumptions about one or more types of user characteristics in models of individual users (e.g. assumptions about their knowledge, misconceptions, goals, plans, preferences, tasks, and abilities);

– the representation of relevant common characteristics of users pertaining to specific user subgroups of the application system (the so-called stereotypes);

– the classification of users as belonging to one or more of these subgroups, and the integration of the typical characteristics of these subgroups into the current individual user model;

– the recording of users' behavior, particularly their past interaction with the system;

– the formation of assumptions about the user based on the interaction history;

– the generalization of the interaction histories of many users into stereotypes;

– the drawing of additional assumptions about the current user based on initial ones;

– consistency maintenance in the user model;

– the provision of the current assumptions about the user, as well as justifications for these assumptions;

– the evaluation of the entries in the current user model, and the comparison with given standards."

This list is of course subject to change when new forms of adaptation to the user require new services from generic user modeling systems. The above list is surprisingly stable though, i.e. it is by and large still valid today. The only important addition would be services to secure users' data and to protect users' privacy (such services are discussed in Section CH3 [51]). In the light of recent progress in the field of recommender systems (see Chapters AT4 [85] and AT6 [14]), services that compare users with other users might also be useful (e.g., delivering a list of nearest neighbors of a given user). In early years researchers tacitly strived for "universal" generic user modeling systems that would ideally perform all important user modeling services. Today, however, a typical generic user modeling system only delivers a small portion of the services listed above, and it is unlikely that this will change very much in the future (see Section 4.5).

3.2 Required Characteristics of Generic User Modeling Systems

Several characteristics of generic user modeling systems have been regarded as very important over the years. We will discuss some of these characteristics in the following. The first two requirements were already proposed very early in the history of generic user modeling systems, while the other requirements became important

only several years ago when commercial generic user modeling systems were developed for use in web site personalization.

Generality, including domain independence. This requirement states that user modeling systems should be usable in as many domains as possible, and within these domains for as many user modeling tasks as possible. While this requirement seemed very important in earlier years, it is less so to date. "Subclasses" of generic user modeling systems have already evolved, most prominently one for student-adaptive tutoring systems that impose very specific requirements on generic student modeling systems (see, e.g., [8, 35, 59, 65, 74] and Chapter MT1 [67]). Such generic student modeling systems are expected to be usable for teaching different subject matters, but not for additional applications besides educational ones.

Expressiveness and strong inferential capabilities. Shell systems and early user modeling servers were expected to be able to express many different types of assumptions about the user at a time. These included beliefs, goals, plans and preferences of the user, as well as all sorts of reflexive assumptions regarding the user and the system (see [47, 92]), and moreover uncertainty and vagueness in these assumptions. Generic user modeling systems were also expected to perform all sorts of reasoning, such as reasoning in a first-order predicate logic, complex modal reasoning (e.g., reasoning about types of modalities), reasoning with uncertainty, plausible reasoning when full information is not available, and to perform conflict resolution when contradictory assumptions are detected.

The rationale for assigning so much importance to expressiveness and strong inferential capabilities lies in the affinity of user modeling research of those days to artificial intelligence, natural-language dialog [58], and intelligent tutoring [39, 89]. User modeling shells were expected to support the complex assumptions and complex reasoning about the user that had been identified in these domains, and additionally to be usable in a wide range of other domains as well. When in the mid-nineties user-adaptive application systems shifted towards different domains with less demanding user modeling requirements (like user-adaptive learning environments that are described in Chapter AP2 [10], and personalized web sites [53]), such highly expressive user modeling and powerful reasoning capabilities became largely redundant.

Support for quick adaptation. In order to bond users with web stores, adaptations should already take place for first-time visitors during their usually relatively short initial interaction. Several commercial user modeling servers can therefore select between more than one modeling and personalization methods with different degrees of complexity, depending on the amount of data that is already available about the user.

Extensibility. Current user modeling servers support a number of user model acquisition and personalization methods, but companies may want to integrate their own methods or third-party tools. Application Programmer Interfaces (APIs) and

interfaces that allow for the (possibly bi-directional) exchange of user information between user-modeling tools are therefore required.

Import of external user-related information. Many businesses already own customer and marketing data, and usually want to integrate these into user modeling systems when starting with personalized e-commerce. To access external data, ODBC interfaces or native support for a wide variety of databases are required. Due to legacy business processes and software, external user-related information often continues to be updated in parallel to the e-commerce application, and therefore needs to be continually integrated at reasonable costs and without impairing the response time.

Management of distributed information. The ability of a generic user modeling system to manage distributed user models is becoming more and more important. Commercial personalized websites often utilize several sources of user information, such as user profiles, purchase records from legacy systems, and customer segmentations from marketing research. Current commercial user modeling servers integrate these information sources already today to a greater or lesser extent [25]. In the future, support for the management of distributed information will also facilitate the integration of mobile user models and of user models in smart appliances (see Sections 4.1 and 4.2).

Support for open standards. Adherence to open standards in the design of generic user modeling systems is decisive since it fosters their interoperability. There already exist efforts in some subfields of user modeling to come up with standards, e.g. for the exchange of user models [29, 30, 56, 60] and for a common user modeling ontology [30, 31, 64, 80], but without much progress so far. Of particular importance for the openness of generic user modeling systems would be the adoption of LDAP [33, 34, 63] which is based on IETF standards. LDAP directories constitute a widely adopted and supported industry standard for storing and retrieving various kinds of people-related information including names, phone numbers, salaries, photographs, digital certificates, passwords, preferences, and even mobile "user agents". Moreover, they support the representation of information about organizations, groups (e.g. administrators) and devices (e.g. printers). Pre-defined ontologies ("schemas") exist for these information types that are easily extensible.

Load balancing. Under real-world conditions, user model servers will experience dramatic changes in their average load. Noticeable response delays or denials of requests should only occur in emergency situations. User modeling servers should be able to react to load increases through load distribution (ideally with CORBA-based components that can be distributed across a network of computers) and possibly by resorting to less thorough (and thereby less time-consuming) user model analyses.

Failover strategies. Centralized architectures need to provide fallback mechanisms in case of a breakdown.

Transactional Consistency. Parallel read/write on the user model and abnormal process termination can lead to inconsistencies that must be avoided by carefully selected transaction management strategies [23]. For instance, if a consistent state cannot be reached, the user model has to be reset to the original state.

Privacy support. Users' privacy concerns, company privacy policies, industry privacy norms, and national and international privacy legislation have a considerable impact on what user-adaptive application may do. Moreover, numerous privacy-enhancing software tools and Internet services are emerging. Generic user modeling systems should also facilitate the compliance with such regulations and user preferences, as well as support privacy-enhancing services. Chapter CH3 [51] discusses privacy-related issues in more detail.

4 Future Trends

It goes without saying that predictions concerning the future of user modeling systems are fairly speculative, due to the rapidly changing nature of application scenarios for personalization and of computing devices that are thereby used. Since personalization has already been demonstrated to benefit both the users and the providers of computer systems and since personalization is therefore likely to stay, it is practically certain that generic tool systems that allow for the easy development and maintenance of personalized systems will be equally necessary in the years to come. The exact shape which future user modeling systems will take on is however likely to be very much contingent on many characteristics of future system usage that are difficult to predict. Here are a few considerations on likely future avenues.

4.1 Mobile User Models

Computing is becoming increasingly mobile and ubiquitous, yet interaction with the user should nevertheless still be personalized. In ubiquitous scenarios, users' handheld devices as well as sensor-equipped environments around them are supposed to provide personalized services wherever users go. In the near future at least, the availability and reliability of mobile networks (and possibly also their bandwidths) is however unlikely to meet the demands of today's client-server architectures for user modeling systems, which require permanent reliable connectivity. Hence mobile user models seem to be worth considering for such scenarios. These user model agents may reside on the server side and be replicated at the beginning of each interaction (see e.g. UMS [24, 52] which is based on LDAP that supports partial or full replication). Or, they may be truly mobile and stay with the user all the time, either on his or her computing device or on an item that the user always wears (like in a wristwatch or jewelry with a wireless connection [26]).

4.2 User Models for Smart Appliances

Personalization has so far been almost exclusively confined to computing systems. Recently, however, appliances are being offered that feature limited but very useful personalization. Examples include car radios with a chip card that contains an authorization code and also stores the driver's preferences concerning pre-set stations, sound volume and tone, and traffic news. Electronic car keys exist that adjust the driver seat, the mirrors and the GPS system to the driver's individual preferences when plugged into the ignition lock. While these are proprietary solutions with proprietary minuscule user models, it is likely that we will see far more examples of personalizable appliances in the future. Since people will not want to carry a small user model gadget for each and every personalized appliance, standardized solutions and hence the need for generic tool systems will soon arise. A particular challenge will be the integration of such local minuscule user models into larger models (e.g. to make them accessible to all cars that the user drives). Generic user modeling systems that support distributed models will have a decisive advantage in this regard.

4.3 Agent-Based User Modeling Systems

The internal structure of user modeling servers (which constitute the currently predominant form of generic user modeling systems) is not transparent to the outside. As was discussed in Section 2.1, various realizations are possible which range from central-repository approaches such as in BGP-MS [55, 78] and Personis [15, 43] to servers that integrate heterogeneous resources of user information (e.g., ATG Adaptive Scenario Engine [3] and enQuire™ Identity Server [20]) and to servers that support the (partial) replication of user modeling server entries (UMS, [24, 52]). These forms of "decentralization" all help increase the performance and failure tolerance of user modeling servers and their ability to integrate into existing environments. Such architectures proved to be able to support personalization even at the largest current websites.

Despite their internal heterogeneity, user modeling servers have however two characteristics in common. For one, they have clearly defined points of access (usually one), at which information about the user and requests for user model entries can be submitted and where answers or spontaneous notifications of the user modeling server can be received. Moreover, even when user modeling servers are decentralized, the interaction of their parts (specifically the flow of control and information) is predetermined by design. Individual parts of user modeling servers are hardly autonomous, i.e. cannot exist without the other parts, and only rarely exhibit "initiative".

Attempts have been made in the past few years to develop new types of architectures for user modeling purposes that do not exhibit these two characteristics. Ideally, such user modeling systems would be conglomerates of independent and autonomous services (or "agents") which collaborate with each other as the need arises (those services may have even been developed by different parties). Decisions on collaboration would not be predetermined by design, but be made dynamically at runtime. A need for such novel kinds of user modeling architectures arose particularly

in the field of ubiquitous computing where numerous small sensors and devices acquire limited information about users and are supposed to exchange it for the provision of personalized services (see Section 4.1). A second area is the support of communities of computer users in which individual users need to contact others who have desired characteristics, e.g. specific knowledge and skills.

Within this broad agent-based paradigm, major differences exist however in the ways in which user models are stored in current research prototypes. In the expert finder system DEMOIR [99], every user possesses a unique user agent. User agents monitor their users and store assumptions about their knowledge and skills. When users need to find experts with certain knowledge and skills, their user agents would broadcast the desired profile to all other user agents, collect their responses, and determine the most similar experts. In a variant architecture, all agents would report (summaries of) their user model contents to a central repository, and contact this repository first when external expertise was sought.

I-Help [95] also features one personal agent per user, which contains a coarse user model that describes predominantly the users' preferences for helping others. I-Help also has other agents, such as diagnostic agents which develop models of the user's knowledge in various domains, and application agents which characterize the services of the applications to which they pertain. Matchmaker agents facilitate locating agents that possess information resources or represent users who are knowledgeable on certain topics. For this purpose, they maintain profiles of the knowledge and some other characteristics of users and applications. When users seek help, their personal agent acts as their representative and negotiates with other agents. In order to do this effectively, user agents also create models of the other agent's "character" and priorities, and collect references to other agents who keep information about their users, for example, diagnostic agents. User model fragments pertaining to a given user may therefore be maintained by several agents in I-Help (in a field study, the authors found up to 20 fragmented models of the same user). These fragments may partially overlap and may even be globally inconsistent.

Even more radical than the mentioned approaches is the work of Lorenz [62] and Specht et al. [90], who envision to break the one-to-one relationship between users and their representing user agents and to replace these agents by a network of small active entities on the client side. In the spirit of self-organizing networks, the authors propose distributed active user modeling agents with extremely limited functionality (which the authors categorize into sensing, modeling, controlling and actuating). The agents would be able to act out of their own initiative and would self-organize appropriately to form "modeling networks". For the communication between the potentially countless tiny agents the authors foresee a mix of the blackboard and the messaging approaches from multi-agent systems [37, 98]. Agents register their services and requests with blackboards provided by local brokers, and brokers broadcast the requests between each other. Communication between different applications would be facilitated by a common user model exchange language (e.g., UserML [29, 30]) and a common user modeling ontology (such as GUMO [30, 31] or that of Razmerita [80]).

Agent-Based User Modeling Systems are currently the object of intensive research and discussion (see, e.g., [19]). One can expect that the same push towards abstraction and more generic architectures that lead to the development of current generic user

modeling systems will take place as soon as these architectures become more homogeneous and more widely adopted, and the requirements from the application side more clearly specified.

4.4 Multiple-Purpose Usage

Information about the characteristics of individual users may not only be interesting for personalization purposes. Other possible applications include organizational directory services, skill inventory systems, and in-house or global expert-finding applications. It seems worthwhile and possible to develop generic user model systems that can support all these different usage purposes for people-related information. This would not only create more powerful generic systems, but could also improve them both from a theoretical and practical perspective. For instance, these new applications may shed new light on the pros and cons of a user model server architecture versus an agent based architecture (see [99]). Or from a more practical perspective, basing generic user modeling systems on LDAP that was developed for directory services (see [24, 52]) would also help elevate these systems from proprietary developments to the realm of open industry standards.

4.5 Different Generic User Modeling Systems will Merrily Co-Exist

A tacit assumption in the 1980s and early 1990s was that only few different user modeling shell systems (and later server systems) would be needed to support the complete range of possible personalized applications. The vast increase of possible scenarios for personalized services with their inherent different demands and constraints has made the likelihood of a very limited number of "user modeling pearl systems" quite unlikely. Instead, one can expect to find a variety of generic user modeling systems, each of which is going to support only few of the very different future manifestations of personalization and other uses of information about users. Privacy requirements, the need to include user information from legacy systems, and the need to exchange user information across different systems will however enforce some standardization, at least at the communication level.

References

1. Allen, J. F.: A Plan-Based Approach to Speech Act Recognition. Dept. of Computer Science, University of Toronto, Canada, Technical Report 131/79, (1979).
2. Allgayer, J., Harbusch, K., Kobsa, A., Reddig, C., Reithinger, N., and Schmauks, D.: XTRA: A Natural-Language Access System to Expert Systems. International Journal of Man-Machine Studies 31, (1989) 161-195, DOI 10.1016/0020-7373(89)90026-6.
3. ATG Adaptive Scenario Engine. (2006), http://www.atg.com/en/products/engine/

4. Billsus, D. and Pazzani, M. J.: User Modeling for Adaptive News Access. User Modeling and User-Adapted Interaction: The Journal of Personalization Research 10, (2000) 147-180, DOI 10.1023/A:1026501525781.

5. Blank, K.: Benutzermodellierung für adaptive interaktive Systeme: Architektur, Methoden, Werkzeuge und Anwendungen. vol. 131. Sankt Augustin, Germany: infix (1996).

6. Brajnik, G. and Tasso, C.: A Shell for Developing Non-monotonic User Modeling Systems. International Journal of Human-Computer Studies 40, (1994) 31-62, DOI: 10.1006/ijhc.1994.1003.

7. Breese, J., Heckerman, D., and Kadie, C.: Empirical Analysis of Predictive Algorithms for Collaborative Filtering. Proc. of the Fourteenth Annual Conference on Uncertainty in Artificial Intelligence (UAI-98), San Francisco (1998) 43-52., ftp://ftp.research.microsoft.com/pub/tr/tr-98-12.pdf.

8. Brusilivsky, P. and Maybury, M.: From Adaptive Hypermedia to the Adaptive Web. Communications of the ACM 45, (2002) 31-33, DOI 10.1145/506218.506239.

9. Brusilovsky, P.: KnowledgeTree: A Distributed Architecture for Adaptive E-Learning. Thirteenth International World Wide Web Conference, WWW 2004 (Alternate track papers and posters), New York, NY (2004) 104-113, 10.1145/1013367.1013386.

10. Brusilovsky, P.: Adaptive Techniques in Web-based Education. In: The Adaptive Web: Methods and Strategies of Web Personalization, Brusilovsky, P., Kobsa, A., and Nejdl, W., Eds. Heidelberg, Germany: Springer Verlag (forthcoming) In this volume.

11. Brusilovsky, P., Kobsa, A., and Nejdl, W.: The Adaptive Web: Methods and Strategies of Web Personalization. Heidelberg, Germany: Springer Verlag (forthcoming).

12. Brusilovsky, P., Sosnovsky, S., and Yudelson, M.: Ontology-based Framework for User Model Interoperability in Distributed Learning Environments. World Conference on E-Learning, E-Learn 2005, Vancouver, Canada (2005) 2851-2855, http://www2.sis.pitt.edu/~peterb/papers/eLearn2005-adapt.pdf.

13. Buchanan, B. G. and Shortliffe, E. H.: Rule-Based Expert Systems: The MYCIN Experiments of the Stanford Heuristic Programming Project. Reading, MA: Addison-Wesley (1984).

14. Burke, R.: Hybrid Web Recommender Systems. In: The Adaptive Web: Methods and Strategies of Web Personalization, Brusilovsky, P., Kobsa, A., and Nejdl, W., Eds. Heidelberg, Germany: Springer Verlag (forthcoming) In this volume.

15. Carmichael, D. J., Kay, J., and Kummerfeld, B.: Consistent Modelling of Users, Devices and Sensors in a Ubiquitous Computing Environment. User Modeling and User-Adapted Interaction: The Journal of Personalization Research 15, (2005) 197-234, DOI: 10.1007/s11257-005-0001-z.

16. Clayton, B. D.: ART Programming Tutorial, Version 1.0. Inference Corporation, Los Angeles, CA. (1985).

17. Cohen, P. R. and Perrault, C. R.: Elements of a Plan-Based Theory of Speech Acts. Cognitive Science 3, (1979) 177-212, DOI 10.1016/S0364-0213(79)80006-3.

18. Identity Management. (2006), http://www.criticalpath.net/en/22/identitymanagement/

19. Dolog, P. and Vassileva, J.: Workshop on Decentralized, Agent Based and Social Approaches to User Modelling (DASUM). 9th International Conference on User Modelling, Edinburgh, Scotland (2005), http://www.l3s.de/~dolog/dasum/DASUM-proceedings.pdf.

20. enQuire Identity Server. Persistent Systems Pvt. Ltd. (2006), http://www.persistentsys.com/products/enquire/enquire.htm

21. Finin, T. W.: GUMS: A General User Modeling Shell. In: User Models in Dialog Systems, Kobsa, A. and Wahlster, W., Eds. Berlin, Heidelberg: Springer-Verlag (1989) 411-430.

22. Finin, T. W. and Drager, D.: GUMS1: A General User Modeling System. Sixth Canadian Conference on Artificial Intelligence, Montreal, Canada (1986) 24-29.

23. Fink, J.: Transactional Consistency in User Modeling Systems. In: UM99 User Modeling: Proceedings of the Seventh International Conference, Kay, J., Ed. Wien New York: Springer-Verlag (1999) 191-200, http://bistrica.usask.ca/UM/UM99/Proc/fink.pdf.

24. Fink, J.: User Modeling Servers - Requirements, Design, and Evaluation. Amsterdam, Netherlands: IOS Press (2004), http://books.google.com/books?q=isbn:1586034057.

25. Fink, J. and Kobsa, A.: A Review and Analysis of Commercial User Modeling Servers for Personalization on the World Wide Web. User Modeling and User-Adapted Interaction: The Journal of Personalization Research 10, (2000) 209-249, DOI: 10.1023/A:1026597308943.

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

27. Gauch, S., Speretta, M., Chandramouli, A., and Micarelli, A.: User Profiles for Personalized Information Access. In: The Adaptive Web: Methods and Strategies of Web Personalization, Brusilovsky, P., Kobsa, A., and Nejdl, W., Eds. Heidelberg, Germany: Springer Verlag (forthcoming) In this volume.

28. Goy, A., Ardissono, L., and Petrone, G.: Personalization in E-Commerce Applications. In: The Adaptive Web: Methods and Strategies of Web Personalization, Brusilovsky, P., Kobsa, A., and Nejdl, W., Eds. Heidelberg, Germany: Springer Verlag (forthcoming) In this volume.

29. Heckmann, D. and Krüger, A.: A User Modeling Markup Language (UserML) for Ubiquitous Computing. In: User Modeling 2003: 9th International Conference, UM 2003, Brusilovsky, P., Corbett, A., and de Rosis, F., Eds. Heidelberg: Springer Verlag (2003) 403-407, http://springerlink.metapress.com/link.asp?id=vnjcm5xr8jhjhvaj.

30. Heckmann, D., Schwartz, T., Brandherm, B., and Kröner, A.: Decentralized User Modeling with UserML and GUMO. Workshop on Decentralized, Agent Based and Social Approaches to User Modelling (DASUM), 9th International Conference on User Modeling, Edinburgh, Scotland (2005) 61-64, http://www.l3s.de/~dolog/dasum/DASUM-proceedings.pdf.

31. Heckmann, D., Schwartz, T., Brandherm, B., Schmitz, M., and von Wilamowitz-Moellendorff, M.: GUMO: The General User Model Ontology. In: User Modeling 2005: 10th International Conference, UM 2005, Edinburgh, Scotland., vol. LNCS 3538, Ardissono, L., Brna, P., and Mitrovic, A., Eds. (2005) 428-432, DOI 10.1007/11527886_58.

32. Herlocker, J., Konstan, J., Borchers, A., and Riedl, J.: An Algorithmic Framework for Performing Collaborative Filtering. Proc. of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval., New York (1999) 230-237, DOI: 10.1145/312624.312682.

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

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

35. Huang, X., McCalla, G. I., Greer, J. E., and Neufeld, E.: Revising Deductive Knowledge and Stereotypical Knowledge in a Student Model. User Modeling and User-Adapted Interaction: The Journal of Personalization Research 1, (1991) 87-115, DOI 10.1007/BF00158953.

36. Intellicorp. http://www.intellicorp.com

37. Jennings, N. R. and Wooldridge, M. J.: Agent Technology : Foundations, Applications, and Markets. Heidelberg, Germany: Springer Verlag (2002).

38. Kass, R.: Acquiring a Model of the User's Beliefs from a Cooperative Advisory Dialog. in Dept. of Information and Computer Science. Philadelphia, PA: University of Pennsylvania (1988).

39. Kass, R.: Student Modeling in Intelligent Tutoring Systems -- Implications for User Modeling. In: User Models in Dialog Systems, Kobsa, A. and Wahlster, W., Eds. Berlin, Heidelberg: Springer-Verlag (1989) 386-410.

40. Kay, J.: UM: A Toolkit for User Modelling. In: Second International Workshop on User Modeling. Honolulu, HI (1990) 1-11.

41. Kay, J.: The um Toolkit for Reusable, Long Term User Models. User Modeling and User-Adapted Interaction: The Journal of Personalization Research 4, (1995) 149-196, DOI: 10.1007/BF01100243.

42. Kay, J.: A Scrutable User Modelling Shell for User-Adapted Interaction. in Basser Department of Computer Science: University of Sydney, Australia (1999), http://www.cs.usyd.edu.au/~judy/Homec/Pubs/thesis.bz2.

43. Kay, J., Kummerfeld, B., and Lauder, P.: Personis: A Server for User Models. In: Adaptive Hypermedia and Adaptive Web-Based Systems: Second International Conference, AH 2002, vol. 2347, *LNCS*, De Bra, P., Brusilovsky, P., and Conejo, R., Eds. Berlin Heidelberg: Springer-Verlag (2002) 203–212, http://springerlink.metapress.com/link.asp?id=2l54yrgc0p8n2d5g.

44. Kleiber, U.: Erklärung in interaktiven Systemen und Unterstützungsmöglichkeiten durch das System BGP-MS. WG Knowledge-Based Information Systems, Department of Information Science, University of Konstanz, Germany, WIS Memo 6, (1994).

45. Knowledge Craft: Knowledge Craft, 3.2 edition. Carnegie Group, Inc., Pittsburgh, PA (1988).

46. Kobsa, A.: Benutzermodellierung in Dialogsystemen. Berlin, Heidelberg: Springer Verlag (1985).

47. Kobsa, A.: A Taxonomy of Beliefs and Goals for User Models in Dialog Systems. In: User Models in Dialog Systems, Kobsa, A. and Wahlster, W., Eds. Berlin, Heidelberg: Springer-Verlag (1989) 52-68.

48. Kobsa, A.: Modeling The User's Conceptual Knowledge in BGP-MS, a User Modeling Shell System. Computational Intelligence 6, (1990) 193-208.

49. Kobsa, A.: Editorial. User Modeling and User-Adapted Interaction 4, (1995) iii-v.

50. Kobsa, A.: Generic User Modeling Systems. User Modeling and User-Adapted Interaction: The Journal of Personalization Research 11, (2001) 49-63, DOI: 10.1023/A:1011187500863.

51. Kobsa, A.: Privacy-Enhanced Web Personalization. In: The Adaptive Web: Methods and Strategies of Web Personalization, Brusilovsky, P., Kobsa, A., and Nejdl, W., Eds. Heidelberg, Germany: Springer Verlag (forthcoming) In this volume.

52. Kobsa, A. and Fink, J.: An LDAP-Based User Modeling Server and its Evaluation. Submitted, (2006).

53. Kobsa, A., Koenemann, J., and Pohl, W.: Personalized Hypermedia Presentation Techniques for Improving Customer Relationships. The Knowledge Engineering Review 16, (2001) 111-155, DOI: 10.1017/S0269888901000108.

54. Kobsa, A., Müller, D., and Nill, A.: KN-AHS: An Adaptive Hypertext Client of the User Modeling System BGP-MS. Proceedings of the Fourth International Conference on User Modeling, Hyannis, MA (1994) 99-105, Reprinted in M. Maybury and W. Wahlster, eds. (1998). Readings in Intelligent User Interfaces. San Mateo, CA: Morgan Kaufman, 372-378. http://www.ics.uci.edu/~kobsa/papers/1994-UM94-kobsa.pdf.

55. Kobsa, A. and Pohl, W.: The BGP-MS User Modeling System. User Modeling and User-Adapted Interaction: The Journal of Personalization Research 4, (1995) 59-106, DOI: 10.1007/BF01099428.

56. A Standard for the Performatives in the Communication between Applications and User Modeling Systems (Draft). (1996), http://www.ics.uci.edu/~kobsa/papers/1996-kobsa-pohl-fink-rfc.pdf

57. Kobsa, A. and Schreck, J.: Privacy through Pseudonymity in User-Adaptive Systems. ACM Transactions on Internet Technology 3, (2003) 149–183, DOI: 10.1145/767193.767196.

58. Kobsa, A. and Wahlster, W.: User Models in Dialog Systems. Berlin: Springer-Verlag (1989).

59. Kono, Y., Ikeda, M., and Mizoguchi, R.: THEMIS: A Nonmonotonic Inductive Student Modeling System. Journal of Artificial Intelligence in Education 5, (1994) 371-413.

60. Kummerfeld, R. and Kay, J.: Remote Access Protocols for User Modelling. Proceedings and Resource Kit for Workshop User Models in the Real World, Chia Laguna, Sardinia (1997) 12-15, http://www.cs.usyd.edu.au/~judy/Homec/Pubs/1997_umnet.html.

61. Liberty Alliance Project: Digital Identity Defined. (2006), http://www.projectliberty.org/

62. Lorenz, A.: A Specification for Agent-Based Distributed User Modelling in Ubiquitous Computing. Workshop on Decentralized, Agent Based and Social Approaches to User Modelling (DASUM), 9th International Conference on User Modeling, Edinburgh, Scotland (2005) 31-40, http://www.l3s.de/~dolog/dasum/DASUM-proceedings.pdf.

63. Loshin, P.: Big Book of Lightweight Directory Access Protocol (LDAP) RFCs. San Diego, CA: Morgan Kaufmann (2000).

64. Learning Technology Standards Committee. (2006), http://ieeeltsc.org/

65. Machado, I., Martins, A., and Paiva, A.: One for All and All in One: A Learner Modelling Server in a Multi-Agent Platform. In: UM99 User Modeling: Proceedings of the Seventh International Conference, Kay, J., Ed. Wien, New York: Springer-Verlag (1999) 211-221.

66. McTear, M.: Special Issue on User Modeling. in Artificial Intelligence Review, vol. 7 (1993).

67. Millan, E. and Brusilovsky, P.: User Models for Adaptive Hypermedia and Adaptive Educational Systems. In: The Adaptive Web: Methods and Strategies of Web Personalization, Brusilovsky, P., Kobsa, A., and Nejdl, W., Eds. Heidelberg, Germany: Springer Verlag (forthcoming) In this volume.

68. Miller, B. N., Konstan, J. A., and Riedl, J.: PocketLens: Toward a Personal Recommender System. ACM Transactions on Information Systems 22, (2004) 437-476, DOI: 10.1145/1010614.1010618.

69. Mobasher, B.: Data Mining for Web Personalization. In: The Adaptive Web: Methods and Strategies of Web Personalization, Brusilovsky, P., Kobsa, A., and Nejdl, W., Eds. Heidelberg, Germany: Springer Verlag (forthcoming) In this volume.

70. Morik, K.: Überzeugungssysteme der Künstlichen Intelligenz: Validierung vor dem Hintergrund linguistischer Theorien über implizite Äußerungen. Tübingen: Niemeyer (1982).

71. Net Perceptions. (2006), http://www.netperceptions.com

72. Orwant, J.: Heterogenous Learning in the Doppelänger User Modeling System. User Modeling and User-Adapted Interaction: The Journal of Personalization Research 4, (1995) 107-130, DOI: 10.1007/BF01099429.

73. Paiva, A. and Self, J.: TAGUS: A User and Learner Modeling System. In: Proc. of the Fourth International Conference on User Modeling. Hyannis, MA (1994) 43-49.

74. Paiva, A. and Self, J.: TAGUS -- A User and Learner Modeling Workbench. User Modeling and User-Adapted Interaction: The Journal of Personalization Research 4, (1995) 197-226, DOI: 10.1007/BF01100244.

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

76. Perrault, C. R., Allen, J. F., and Cohen, P. R.: Speech Acts as a Basis for Understanding Dialogue Coherence. Department of Computer Science, University of Toronto, Canada, Report 78-5, (1978).

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

78. Pohl, W.: Logic-Based Representation and Reasoning for User Modeling Shell Systems. User Modeling and User-Adapted Interaction: The Journal of Personalization Research 9, (1999) 217-282, DOI 10.1023/A:1008325713804.

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

80. Razmerita, L., Angehrn, A., and Maedche, A.: Ontology-Based User Modeling for Knowledge Management Systems. In: User Modeling 2003: 9th International Conference, UM 2003, Brusilovsky, P., Corbett, A., and De Rosis, F., Eds. Heidelberg, Germany: Springer Verlag (2003) 213-217, http://springerlink.metapress.com/link.asp?id=thw9rmvmvklx9hac.

81. Rich, E.: Building and Exploiting User Models. in Department of Computer Science. Pittsburgh, PA: Carnegie-Mellon University (1979).

82. Rich, E.: User Modeling via Stereotypes. Cognitive Science 3, (1979) 329-354.

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

84. Rich, E.: Stereotypes and User Modeling. In: User Models in Dialog Systems, Kobsa, A. and Wahlster, W., Eds. Berlin, Heidelberg: Springer (1989) 35-51.

85. Schafer, J. B., Frankowski, D., Herlocker, J., and Sen, S.: Collaborative Filtering Recommender Systems. In: The Adaptive Web: Methods and Strategies of Web Personalization, Brusilovsky, P., Kobsa, A., and Nejdl, W., Eds. Heidelberg, Germany: Springer Verlag (forthcoming) In this volume.

86. Schreck, J.: Security and Privacy in User Modeling. Dordrecht, Netherlands: Kluwer Academic Publishers (2003), http://www.security-and-privacy-in-user-modeling.info.

87. Shortliffe, E. H.: Computer-Based Medical Consultations: MYCIN. New York: North-Holland (1976).

88. Sleeman, D.: UMFE: A User Modelling Front-End Subsystem. International Journal of Man-Machine Studies 23, (1985) 71-88.

89. Sleeman, D. and Brown, J. S.: Intelligent Tutoring Systems. New York: Academic Press (1982).

90. Specht, M., Lorenz, A., and Zimmermann, A.: Towards a Framework for Distributed User Modelling for Ubiquitous Computing. Workshop on Decentralized, Agent Based and Social Approaches to User Modelling (DASUM), 9th International Conference on User Modeling, Edinburgh, Scotland (2005) 31-40, http://www.l3s.de/~dolog/dasum/DASUM-proceedings.pdf.

91. Sun: Sun Java System Identity Manager. (2006), http://www.sun.com/software/products/identity_mgr/.

92. Taylor, J. A., Carletta, J., and Mellish, C.: Requirements for Belief Models in Cooperative Dialogue. User Modeling and User-Adapted Interaction: The Journal of Personalization Research 6, (1996) 23-68, DOI 10.1007/BF00126653.

93. IBM Tivoli Identity Manager. (2006), http://www-306.ibm.com/software/tivoli/products/identity-mgr/

94. van Melle, W.: System Aids in Constructing Consultation Programs: EMYCIN. Ann Arbor, MI: UMI Research Press (1982).

95. Vassileva, J., McCalla, G., and Greer, J.: Multi-Agent Multi-User Modeling in I-Help. User Modeling and User-Adapted Interaction: The Journal of Personalization Research 13, (2003) 179-210, DOI: 10.1023/A:1024072706526.

96. Vergara, H.: PROTUM: A Prolog Based Tool for User Modeling. WG Knowledge-Based Information Systems, Department of Information Science, University of Konstanz, Germany, WIS-Report 10, (1994).

97. Wahlster, W. and Kobsa, A.: User Models in Dialog Systems. In: User Models in Dialog Systems, Kobsa, A. and Wahlster, W., Eds. Heidelberg – Berlin: Springer Verlag (1989).

98. Wooldridge, M. and Jennings, N.: Intelligent Agents: Theory and Practice. The Knowledge Engineering Review 10, (1995) 115-152.

99. Yimam, D. and Kobsa, A.: Expert Finding Systems for Organizations: Problem and Domain Analysis and the DEMOIR Approach. In: Beyond Knowledge Management: Sharing Expertise, Ackerman, M., Cohen, A., Pipek, V., and Wulf, V., Eds. Cambridge, MA: MIT Press (2003), http://www.ics.uci.edu/~kobsa/papers/2003-JOCEC-kobsa.pdf.