Peter Dolog and Julita Vassileva (Eds.)

Decentralized, Agent Based and Social Approaches to User Modelling

Workshop DASUM-05,

9th International Conference on User Modelling, UM'2005, Edinburgh, 25 July 2005

Proceedings

Preface

Time is ripe to discuss decentralized approaches to user modelling, since decentralized applications are becoming prevalent both in web-based and mobile/ubiquitous environments. Such applications include personal guides or helpers for navigation or ambient devices, integrated web-sites (e.g. newspapers or magazines), portals (e.g. Yahoo), e-commerce web-sites (e.g. Amazon, e-Bay), and recommender sites. Decentralized applications are loosely coupled systems, usually based on Web Service(s), and can be considered as conglomerates of independent, autonomous services, multi-agent systems, developed by independent parties, which have not been integrated by design, but integrate dynamically at run-time, as the need arises. For example, e-learning courses assembled dynamically from independently created repositories of learning objects and tailored to the needs of a particular learner. A frequently used metaphor is a free-market of services where the user is a shopper that buys a larger service composed dynamically by smaller services.

In industry, decentralized approaches are being deployed for customer profile exchange or permission-based approaches like P3P where data is collected at the user's side and access to these data is under the user's control. Another area where decentralized and social approaches to user modelling are applied is the enrichment of profile data (obtained from whatever source) by information from completely unrelated sources, for example, demographic and sociographic data. An example for this is the Lifestyle Finder and its method of demographic generalization. In ubiquitous environments distributed sensors follow a user’s movements and based on the user’s typical tasks or preferences learned from history and context features, appropriate adaptations of the interface, ambient features, or functionality are made. Decentralized user modeling in this application domain studies how to combine fragmentary / episodic user data and make most sense of it in the specific context with limited processing resources.

The problem of user adaptation is present in both the integrated design (e.g. current recommender systems) and the loosely coupled systems. But while the integrated design systems can rely on centralized user model servers, the market-like ones can not, since the centralized UM model is too restrictive. It imposes a list of user features that it can represent and a non-negotiable format of representation, APIs, and protocol. It also introduces a central point

of failure, and while reliability can be increased by introducing mirrors or distributing the load on several servers, the problem of synchronization and coordination of the mirror servers increases the cost. For large websites like eBay and Amazon, this may not be a problem, but for small players it may be too expensive.

An alternative approach is to allow small players (e.g. individual or clusters of web-services, agents, individual networked applications) to share user information. For this it is necessary to develop decentralized approaches for user modelling.

In decentralized settings, each small player (agent, smart sensor, mobile device, learning object, application, web-service) maintains a small user model / profile, as needed for its own purposes of adaptation. These models are updated by the players sporadically, whenever they interact with users. However, the players can also talk with other players who interact with users and build their own models to exchange user information, and to be more up-to-date. In this way, through communication, the agents leverage the benefits of the UM efforts done by many modellers. Instead of one central UM acting as a sink where the subscribed applications report their user data, or instead of having isolated UMs for each application, we have a community of adaptive applications sharing user information.

There are many questions that arise:

- How do we define a (a decentralized) user model? There is no central notion of a model, but user data fragments dispersed among the various services and agents; the level of this data can be hugely different, from usage statistics applied by recommender systems, to detailed preference models computed by negotiation agents. Do we talk of all these as "models" or do we define the model in relation to the purpose for which it will be used?
- How to ensure that different autonomous, independently created applications and services are able to communicate with each other and exchange user data, if each represents its user models in a different way (ontologies, communication protocols and languages).
- Can user data harvested in one context be useful for adaptation in another context? How to re-compute the collected user data in a user model to be used the new context? Should the little player do this computation or special components that know what user data is relevant for what type of adaptation? (architectures for adaptation, standard user modelling tasks / purposes).

- How to represent the knowledge used in computing user models on demand, the knowledge about what data is relevant and how to use it to adapt a given service? Should representations be centered around the user data, the content, or around the purposes of use? Should they be declarative or procedural? Is any data ever irrelevant? What user data to keep and what not to keep after adaptation? (knowledge representation, reasoning, learning).
- Whom to ask for user information? Who is "trusted"? How to define trust in this case: an agent that is honest, or similar to the client agent in its criteria, or both? How to use models of interpersonal relationships and how to interpret user data received from acquaintances? (trust, reputation, gossiping)
- How to ensure privacy? - Whose responsibility is it? Who "owns" the data - the user or the service? (transactional data is normally "owned" by both parties) How do we regulate the propagation of user data between applications, with respect to the user's privacy (e.g., P3P policies, ...)
- What are the candidate applications: where do we expect to see such applications first? Recommender systems? Mobile / ubiquitous computing applications or ambient computing environments? E-learning systems? E-commerce systems? What is common among these application areas?
- How to resolve conflicts between user model fragments kept by different agents (e.g. using information about adaptation and purpose)?

The workshop attracted a relatively high number of submissions (19) of which 6 full papers (31.5%) and 6 short papers were accepted for presentation at the workshop.

Several of the accepted papers address approaches for assembling user model fragments or “modules” using information about the purpose of adaptation (Niu et al.), the user’s task and role (Lock & Kudenko) and the user’s social networks (Gonzalez et al.).

The paper by Whitaker and Kay addresses two interesting questions in decentralized user modeling which are relevant for several other papers in the workshop. The first one is how to identify users; how to be sure that different pieces of data/evidence received from different sensors (modelers), or the same sensor at different times are related to the same user? Whitaker and Kay propose an architecture for centralized user modeling in ubiquitous environments which provides one possible solution of this problem. Decentralized user modeling architectures for ubiquitous computing environments are proposed by Lorenz & Specht, Lorenz & Zimmermann and Heckmann et al.

The second question is how to cluster users into groups and how to associate an individual with a group. Since user models can contain a variety of characteristics, deciding which characteristics should become a basis for clustering depends on many factors. Berkovsky et al. present a multi-agent framework for distributed collaborative filtering in which they experiment with two possible approaches for grouping users: based on geographical proximity and based on topic proximity. They propose also a third approach, based on demographic proximity (which they call "social filtering"). Their experiments show that using topic proximity groups outperforms geographical proximity and that collaborative filtering based on geographical proximity can be improved by augmenting it with demographic proximity. In the conclusions, the authors raise an important question that applies to not just collaborative filtering, but to decentralized user modeling system in general – why would repositories (or user modelers) cooperate and share user data. In a competitive environment, there may be no incentive for sharing. However, there is already a market for user data (e.g. e-mail addresses etc.). We can imagine further fragmentation and automatisation of this market where sensors, agents, applications, and services negotiate and share user data fragments. As in any electronic market system, issues of security, reputation, trust and privacy arise. Two of the articles (Olorunleke & McCalla, Regan et al.) discuss such issues.

Finally, an important question is how to combine different user models if we want to build group models for computer-supported collaborative system. What additional social factors, apart from the models of the individuals that participate in the group need to be accounted for? Wilson's paper suggests that modeling the (task-external) status of the members in a group can be an important factor in the deliberation process in small groups.

With this collection of papers, we believe there would be a ground for interesting discussions and some integration of ideas at the workshop. We would like to thank all the authors for submitting their papers to the workshop and especially to the program committee members who provided critical and constructive feedback in reviewing the papers.

Peter Dolog and Julita Vassileva

Workshop Chairs:

Peter Dolog
L3S Research Center,
University of Hannover,
Expo Plaza 1, 30539 Hannover
Germany

Julita Vassileva
Computer Science Department
University of Saskatchewan
Saskatoon, SK, S7N 5C9
Canada

Program Committee:

- Liliana Ardissono, University of Torino, Italy
- Lora Aroyo, Eindhoven University of Technology, The Netherlands
- Mathias Bauer, DFKI, Germany
- Peter Brusilovsky, University of Pittsburgh, PA, USA
- Susan Bull, University of Birmingham, UK
- Keith Cheverst, Lancaster University, UK
- Robin Cohen, University of Waterloo, Canada
- Nadia de Carolis, University of Bari, Italy
- Ludger van Elst, DFKI, Germany
- Elena Gaudioso, University for Distance Learning, Spain
- Piotr Gmytrasiewicz University of Illinois, Chicago, USA
- Dominik Heckmann, DFKI, Germany
- Judy Kay, University of Sydney, Australia
- Alfred Kobsa, University of California at Irvine, USA
- Antonio Krueger, DFKI, Germany
- Daniel Kudenko, University of York, UK
- Gord McCalla, University of Saskatchewan, Canada
- Daniel Olmedilla, L3S Research Center, University of Hanover, Germany
- Olayide Olorunleke, University of Saskatchewan, Canada
- Fiorella de Rosis, University of Bari, Italy
- Boris de Ruyter, Philips Research, The Netherlands
- Michael Sintek, DFKI, Germany
- Amy Soller, Institute for Defence Analyses, USA
- Thomas Tran, University of Ottawa, Canada
- Marianne Winslett, University of Illinois at Urbana-Champaign, USA

TABLE OF CONTENTS

(in alphabetical order by the first author's family name)

Full Papers

Short Papers

Collaborative Filtering over Distributed Environment

Shlomo Berkovsky[1], Paolo Busetta[2], Yaniv Eytani[1], Tsvi Kuflik[3], Francesco Ricci[2]

[1] University of Haifa, Computer Science Department,
{slavax, ieytani}@cs.haifa.ac.il

[2] ITC-irst, Trento
{busetta,ricci}@itc.it

[3] University of Haifa, Management Information Systems Department
tsvikak@is.haifa.ac.il

Abstract. Currently, implementations of the Collaborative Filtering (CF) algorithm are mostly centralized. Hence, information about the users, for example, product ratings, is concentrated in a single location. In this work we propose a novel approach to overcome the inherent limitations of CF (sparsity of data and cold start) by exploiting multiple distributed information repositories. These may belong to a single domain or to different domains. To facilitate our approach, we used LoudVoice, a multi-agent communication infrastructure that can connect similar information repositories into a single virtual structure called "implicit organization". Repositories are partitioned between such organizations according to geographical or topical criteria. We employ CF to generate user-personalized recommendations over different data distribution policies. Experimental results demonstrate that topical distribution outperforms geographical distribution. We also show that in geographical distribution using filtering based on social characteristics of the users improves the quality of recommendations.

1 Introduction

Collaborative Filtering (CF) [5] is commonly used in many E-Commerce recommender systems to support users selecting music CDs, movies, and more [17]. CF is based on the assumption that people with similar tastes prefer the same items. In order to generate a recommendation, CF initially creates a neighborhood of users with the highest similarity to the user whose preferences are to be predicted. It then generates a prediction by calculating the normalized and weighted average of the ratings of the users in the neighborhood.

The input for the CF algorithm is a model of the user, i.e., information describing the user's preferences (interests, habits, and so on) in the form of a feature vector. This vector is matched against all other users' vectors, and k most similar users are selected to generate a recommendation. State of the art CF systems usually collect user models by tracking the users' past interactions with the systems, and storing this information in their local repositories. CF systems are known to suffer from two

inherent drawbacks [3]: sparsity (lack of sufficient information about the users) and cold-start (no information about a new user or item recently added to the system).

In real life conditions, information about the users is naturally distributed among many data repositories, in a variety of domains. When integrated, these repositories could provide a recommendation, while a CF based on a single repository may fail to do so. In this work we discuss the details of operating CF over a distributed setting of data repositories and compare different distribution approaches. We facilitated the development of the above ideas using LoudVoice infrastructure. LoudVoice supports group communication in multi-agent systems, where similar service-providing agents are connected into a single virtual structure called "implicit organization" [1].

To evaluate the feasibility of our approach, we conducted several experiments. We measured the impact of different data distribution scenarios on the quality of recommendations. We compared two types of distribution representing possible real-life conditions:

- Geographical distribution - imitates a situation where information about a particular user is available only in his close vicinity. In this scenario, each LoudVoice organization represents a limited geographical area.
- Topical distribution – imitates a situation where each repository stores information related to a limited number of topics (objects types).

Experimental results show that the topical criterion is superior to the geographical criterion. Additional experiments demonstrate that applying CF using social distinction considerations (such as age, occupation and gender) improves the quality of recommendations.

The rest of the paper is organized as follows. Section *2* reviews the related works on distributed Collaborative Filtering and discusses the details of LoudVoice communication infrastructure. Section *3* discusses the possible policies of data distribution and the details of CF over the distributed environments. Section *4* presents the details of distributed CF implementation over LoudVoice. Section *5* presents experimental results. Finally, we conclude and present the directions of future research.

2 Distributed Collaborative Filtering

Collaborative filtering is probably the most familiar, most widely implemented, and most mature recommendation technique. It relies on the idea that people who agreed in the past will also agree in the future [18].

The input for the CF recommender system is a matrix of user ratings for items, where each row represents the ratings of a single user and each column represents the ratings for a single item. CF aggregates ratings of items to recognize similarities between users, and generates a new recommendation of an item by weighting the ratings of similar users for the same item [4]. The main advantage of CF is that it is completely independent of any item representation. Thus items can be recommended regardless of their contents.

2.1 Related Works

Implementing the CF algorithm in a decentralized way was initially proposed in [19]. It presents a Peer-to-Peer architecture supporting product recommendations for mobile customers represented by software agents. The communication between the deployed agents used an expensive routing mechanism based on network flooding that increased the communication overhead. An improved mechanism was proposed in [11]; however, it reduced the efficiency of the neighborhood formation phase. The work in [16] elaborated on the discussion of distributed CF. It developed a detailed taxonomy of distributed CF in recommender systems and presented different implementation frameworks for different domains of Electronic Commerce. Most of these studies did not include thorough experimentation and did not analyze the different factors that might affect the quality of the generated recommendations.

The PocketLens project [10] implemented and compared five distributed architectures for CF. It was found that no architecture is perfect, but the performance of a content-addressable mechanism [15] is close to that of a centralized CF algorithm, while the encrypted communication protocol [2] can add the essential dimension of security.

Privacy is an inherently related to the issue of decentralized distribution. In a decentralized setting the information resides on the client-side. Thus individual users might restrict access to the information by deciding which other users are authorized to receive their personal information. P3P privacy policies [13], and also the work reported in [8], address the privacy issue. It is suggested that access to the information repositories should be restricted, decreasing the likelihood of information concerning a given user being overheard by undesired parties.

Other works propose a multi-agent approach to control, and filter access to the data, depending on the user role [7], to improve privacy preservation by forming user communities. These communities acquire an encrypted aggregate user profile, representing the group as whole and not individual users [2], and employ randomized perturbation to obfuscate sensitive information about the users [14] and to minimize the possibility of acquiring such information through malicious attacks.

2.2 Self-Organized Communication Platform

In this work we employ CF over a set of distributed data repositories, where each repository acts as an independent agent. In order to minimize communication overheads, we require a platform that supports a method of communication between the relevant agents only.

LoudVoice is an efficient multi-agent communication platform based on the concept of channeled multicast [1]. Messages are sent on a channel and received by all agents that “tune” into it. Channeled multicast reduces the amount of communication needed when more than two agents are involved in a task, and allows overhearing, i.e., the ability to listen to messages addressed to others. Overhearing, in turn, enables functionality such as the collection of contextual information, pro-active assistance, and monitoring without interfering with the existing protocols.

LoudVoice has been designed to support the notion of implicit organizations. An implicit organization is a group of agents playing the same role on a given channel and willing to coordinate their actions for the sake of delivering a service. The term "implicit" highlights the fact that there is no need for a group formation phase, since joining an organization is a matter of tuning into a channel. By definition, implicit organizations are formed by agents able to play the same role. LoudVoice allows senders to address messages either to specific agents or to all agents that offer a certain service on a channel, for example providers of a particular type of information.

3 Distribution of Repositories

The neighborhood formation phase in CF finds a set of users who are similar to the user whose prediction is generated (the active user). Traditional centralized implementations typically require computing similarity between the active user and every other user in the system for the purpose of finding the set of the *K* most similar users. In a distributed environment, information about users is partitioned among different repositories. Computing similarity between users requires information stored in different and remote repositories to be combined.

In the following sub-sections we analyze two conceptually different policies for partitioning the data between the various repositories, and discuss the implications with regard to the phase neighborhood formation.

3.1 Geographical Distribution

A natural form of data distribution is "geographical distribution", where information about users is available only in their physical vicinity. For example, the reading preferences of a user are usually found in his/her local (and only local) library or bookstore. We can assume that the set of items rated by all users, in different geographical locations, is roughly similar. As each repository contains the ratings of a subset of users, geographical distribution is virtually a horizontal partitioning of the ratings matrix. Thus, the phase of neighborhood formation must comprise a search for similar users in all the repositories.

In this distribution, the set of rated items (by all users) in different repositories is fundamentally identical, and all information about a particular user is concentrated in a single repository. Therefore, to compute the similarity, the set of all the rated items of the active user should be sent to the remote repositories. Each remote repository locally computes the similarity between the active user and each of the locally stored users, and returns a "local" neighborhood. Thus, a "global" neighborhood for the active user is generated by combining all the sets of previously formed "local" neighborhoods and re-ranking the resulting set according to the users' similarity to the active user.

Comparing to a centralized CF algorithm, forming the neighborhood over geographical distribution of data repositories spreads computational load between the repositories. This occurs as each repository locally "eliminates" a portion of globally

dissimilar users, thus reducing the computational complexity of the combination of "local" neighborhoods.

In addition, geographical distribution enhances the privacy of CF, as the ratings of the users (except those of the active user) are exposed only within the boundaries of the repository (that is assumed to be more secure). Instead of all the ratings, only similarity values are transferred over the network.

3.2 Topical Distribution

A different form of data repository distribution is achieved by considering the variety of diverse domains of items (books, music, movies, and so on). This is referred to as "topical distribution" and can be considered as a vertical partitioning of the rating matrix. Each repository stores the ratings for items related to one particular domain. Thus, sets of items stored in different repositories do not completely overlap. Relying on a single domain for finding similar users might prove insufficient and might require constructing a global view of a user's ratings by combining information from the remote repositories.

In topical distribution, sets of rated items in various repositories may be different and the information about a particular user is divided among multiple repositories. Thus, there is no sense in sending the ratings of the active user items to the remote repositories, as these items might not be found there. Therefore, we base the neighborhood formation phase on the globally unique identifier (called *user-id*) of the active user (assuming that the user registers into different systems with this unique identifier only, and that the remote repository may have served the user in previous sessions).

To find the set of similar users, the active user's *user-id* is transferred over the network to the remote repositories. Each repository computes the "local" neighborhood according to ratings of its own stored items, and returns *user-ids* of potentially similar users. Similarity between the active user and the users, whose *user-ids* were returned by the remote repositories, is computed locally in the repository that initiated the recommendation process. Finally, K most similar users form the neighborhood. This type of neighborhood formation is based on the observation that the most similar users are similar in many domains and thus in a number of data repositories.

Topical distribution also enhances the privacy aspects of CF algorithm, as only local ratings are needed to compute similarity. No ratings (even those of the active user) are transferred over the network, only the *users-ids*.

3.3 Social Pre-Filtering

In addition to applying various data distribution policies, other factors could be considered for both generating a smaller neighborhood and achieving a more accurate prediction. Such considerations are based on social factors. For example, forming a "local" neighborhood in geographical distribution might limit the similarity

computations to the subset of users that match the active user in one of the social criteria, such as age, occupation, social status, and so on.

Such approximation methods pose a tradeoff. On the one hand, they pre-filter potentially similar users and decrease the amount of available users for the neighborhood formation phase, thus increasing the influence of possible noise in the data. On the other hand, they might improve the accuracy of the CF, as they tend to limit the set of potentially similar users to the set of candidate users whose values for important properties are similar to those of the active user. This is actually coincides with the general notion of CF, the basis of which is a search of similar users for the purposes of building an accurate prediction.

4 Implementation details

We implemented both the topical and geographical data partitions according to the approaches presented in the previous section. We chose LoudVoice [1] to serve as an underlying communication platform. LoudVoice is an appropriate platform due to its channeled multicast capability, discussed earlier in section 2.2. This capability allowed us to handle distribution of data repositories easily, to base the distributed implementation of CF on a standardized API, and to minimize the communication overheads tied to the distribution.

Each LoudVoice channel (communication line) potentially contains a set of data repositories of either the same domain or close geographical vicinity. These data repositories are considered as an "implicit organization" of repositories. Each data repository is represented on a channel by a designated agent, whose role is to allow communication with the other agents on the channel. Organization of data sources is achieved by assigning each agent to a set of relevant LoudVoice channels, reflecting the type of the partitioning.

In addition to the agents representing data repositories, one arbitrary agent is connected to each channel and serves as a "mediator" vis a vis the other channels. This agent is connected both to his original channel and to the inter-organization communication channel. The mediators transfer requests and responses between different channels. For example, consider the structure of two LoudVoice channels "channel A" and "channel B" as illustrated in figure *1*.

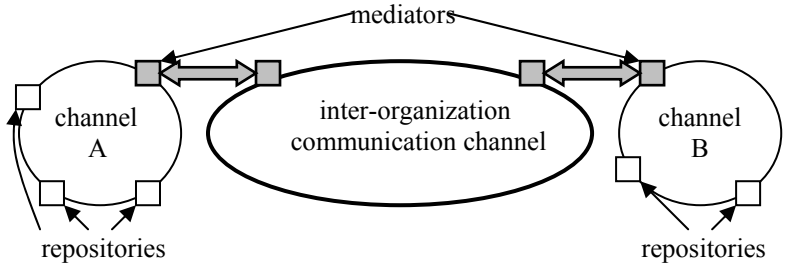

Figure 1, System Architecture over LoudVoice

5 Experimental Results

Experiments were conducted using the publicly available "1 Million Ratings" MovieLens data set [12]. It contains over one million ratings of more than *6000* different users for approximately *4000* different movies.

The first experiment aims at testing the effect of partitioning the data among multiple repositories on the quality of produced recommendations. The data was partitioned both according to geographical distribution (thus, the set of rated items in each repository is identical), and topical distribution (thus, the set of rated items might be different in different repositories).

The MovieLens dataset was partitioned among a gradually increasing number of repositories. For each number of repositories a *90%* subset of the available movie ratings was chosen to be the training set of the CF, and predictions were generated for the remaining *10%* of the ratings. The accuracy of the prediction was measured by comparing the generated prediction with the real ratings found in the data set. The metrics for the accuracy of the prediction was Mean Average Error (MAE) [6] that was computed by:

$$MAE = \frac{\sum_{i=1}^{N} | p_i - r_i |}{N},$$

where N denotes the total number of the predicted items, p_i is the predicted, and r_i is the real rating on item i. To obtain statistic significance, the experiments were repeated *10* times for each number of repositories. Figure *2* illustrates the average values of MAE as a function of the number of data repositories.

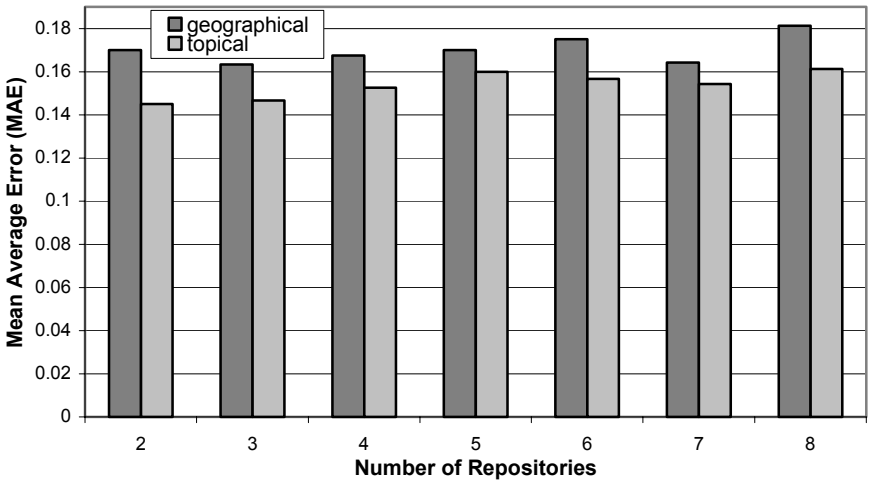

Figure 2, MAE vs. the number of repositories

The figure shows the MAE as a function of the number of repositories for both geographical (left column) and topical (right column) distributions. The MAE values are relatively low, approximately *0.14 – 0.18*, implying that generated predictions are close to the real ratings and that the MAE values are roughly indifferent to the number of repositories. The MAE values measured in the experiments are similar to those obtained in previous studies using the MovieLens dataset (initially presented in [4], and recently compared in [9]).

A comparison of the two above types of distribution shows that for any given number of repositories topical distribution slightly outperforms geographical distribution. This indicates that when the similarity is computed based only on a smaller set of relevant items, the resulting neighborhood is "closer", and as a result, the generated prediction is more accurate.

The goal of the second experiment was to measure the gains in accuracy achieved by applying social pre-filtering in addition to the geographical distribution policy. In each experiment social pre-filtering was based on one of the following social characteristics of the users: age, occupation, or gender. This information was extracted from the basic social data of the users, provided by the MovieLens dataset.

We partitioned the MovieLens dataset among a gradually increasing number of repositories. For each number, we operated each time one of the above social pre-filtering criteria: age, occupation, or gender. In this experiment also a *90%* subset of the available movie ratings was chosen to be the training set of the CF, and the predictions were generated for the remaining *10%* of the ratings. The list of potentially similar users was filtered by computing the similarity only for the users that matched the active user in the relevant social criterion. Accuracy of the prediction was computed using the MAE metrics. Figure *3* illustrates the MAE results as a function of the number of repositories.

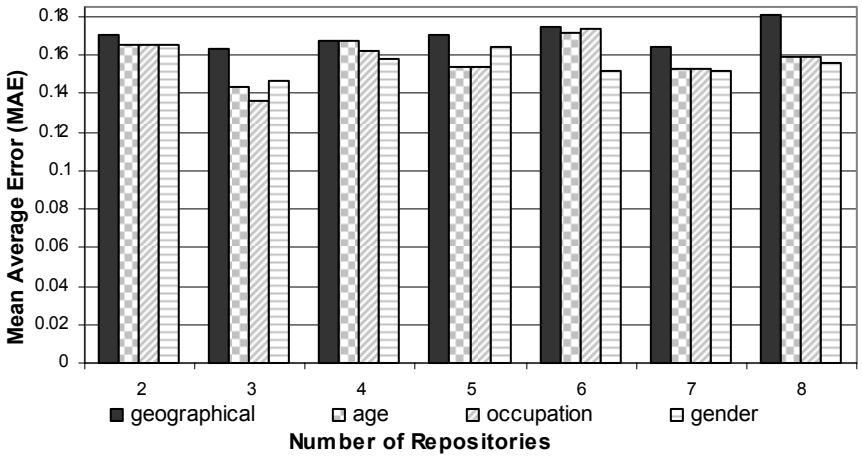

Figure 3, MAE vs. the number of repositories

Figure *3* shows that using social pre-filtering improves the MAE values, in comparison to the regular geographical distribution, although not drastically. When using the age or occupation criterion, the remaining sets of potentially similar users are relatively small. This magnifies the possible influence of noise in the data. We noticed that gender-based social pre-filtering does not act in a consistent way. Thus, experimental evidence shows that social pre-filtering generally improves prediction accuracy. However, we could not currently identify a single most contributing criterion.

6 Conclusions and Future Research

This work demonstrates the possibility of performing collaborative filtering (CF) over a distributed set of data repositories in order to resolve CF's sparsity and cold-start problems. We propose and analyze different policies for the distribution of repositories (topical and geographical partitioning). We also discuss the implementation details for each form of distribution. We suggest that preliminary filtering, based on the users' social characteristics, should be applied to improve the accuracy of the distributed CF. Though this work does not directly deal with privacy enhancement of the CF process, the proposed method of data distribution inherently contributes to solving some of the privacy issues.

The experimental results show that the accuracy of the prediction obtained from distributed CF is similar to the accuracy of state-of-the-art centralized CF systems. A comparison of two distribution policies shows that topical distribution slightly outperforms geographical distribution (regardless of the number of repositories). When the CF is preceded by social pre-filtering, the prediction accuracy increases.

A major issue that is not in the scope of this work is possible commercial competition in the E-Commerce realm. This could hamper performance by limiting cooperation and data sharing between various repositories. In the future, we plan to develop a generic model for users' cooperation and information trading.

In addition, several issues need to be addressed, such as the assumption that users' similarity remains across different organizations, and the fact that even within the same organizations terminology used by different service providers and users might differ and some kind of translation mechanism might be needed.

References

[1] P. Busetta, A. Dona, M. Nori, "*Channeled Multicast for Group Communications*", in Proceedings of the International Joint Conference on Autonomous Agents and Multi-Agent Systems, Bologna, Italy, 2002.

[2] J. Canny, "*Collaborating Filtering with Privacy*", in Proceedings of IEEE Conference on Security and Privacy, Oakland, CA, 2002.

[3] N. Good, J. B. Schafer, J. A. Konstan, A. Borchers, B. Sarwar, J. Herlocker, J. Riedl, "*Combining Collaborative Filtering with Personal Agents for Better Recommendations*",

in Proceedings of the National Conference of the American Association of Artificial Intelligence, Orlando, FL, 1999.

[4] J. L. Herlocker, J. A. Konstan, A. Borchers, J. Riedl, "An Algorithmic Framework for Performing Collaborative Filtering", in Proceedings of the International SIGIR Conference on Research and Development in Information Retrieval, Berkeley, CA, 1999.

[5] J. Herlocker, J. A. Konstan, J. Riedl, "*Explaining Collaborative Filtering Recommendations*", in Proceedings of ACM Conference on Computer Supported Cooperative Work, Philadelphia, PA, 2000.

[6] J. L. Herlocker, J. A. Konstan, L. G. Terveen, J. T. Riedl, "*Evaluating Collaborative Filtering Recommender Systems*", in ACM Transactions on Information Systems, Vol. 22(1), pp. 5-53, 2004.

[7] L. Kagal, T. Finin, A. Joshi, "*A Policy Based Approach to Security for the Semantic Web*", in Proceedings the International Semantic Web Conference, Sanibel Island, FL, 2003.

[8] A. Kobsa, J. Schreck "*Privacy through Pseudonymity in User-Adaptive Systems*", in ACM Transactions on Internet Technology, Vol.. 3(2), pp. 149-183, 2003.

[9] D. Lemire, A. Maclachlan, "*Slope One Predictors for Online Rating-Based Collaborative Filtering*", in proceedings of the SIAM Data Mining Conference, Newport Beach, CA, 2005.

[10] B. N. Miller, J. A. Konstan, J. Riedl, "*PocketLens: Toward a Personal Recommender System*", in ACM Transactions on Information Systems, Vol. 22 (3), 2004.

[11] T. Olsson, "*Decentralised Social Filtering based on Trust*", in Proceedings of the National Conference of the American Association of Artificial Intelligence Recommender Systems Workshop, Madison, WI, 1998.

[12] "One Million MovieLens Dataset", http://www.grouplens.org/data/million/

[13] "P3P Public Overview", http://www.w3.org/P3P/

[14] H. Polat, W. Du, "*Privacy-Preserving Collaborative Filtering Using Randomized Perturbation Techniques*", in Proceedings of International Conference on Data Mining, Melbourne, FL, 2003.

[15] S. Ratnasamy, P. Francis, M. Handley, R. Karp, S. Shenker, "*A Scalable Content-Addressable Network*", in proceedings of ACM SIGCOMM Conference, San Diego, CA, 2001.

[16] B. M. Sarwar, J. A. Konstan, J. Riedl, "*Distributed Recommender Systems: New Opportunities for Internet Commerce*", a chapter in "Internet Commerce and Software Agents: Cases, Technologies and Opportunities", Idea Group Publishers, 2001.

[17] J. B. Schafer, J. A. Konstan, J. Riedl, "*E-Commerce Recommendation Applications*", in Journal of Data Mining and Knowledge Discovery, Vol. 5 (1/2), pp. 115-152, 2001.

[18] U. Shardanand, P. Maes, "*Social Information Filtering: Algorithms for Automating "Word of Mouth"*", in Proceedings of the International Conference on Human Factors in Computing Systems, Denver, CO, 1995.

[19] A. Tveit, "*Peer-to-Peer Based Recommendations for Mobile Commerce*", in Proceedings of the International Workshop on Mobile Commerce, Rome, Italy, 2001.

Smart User Models: Modelling the Humans in Ambient Recommender Systems

Gustavo González[1], Cecilio Angulo[2], Beatriz López[1], Josep Lluís de la Rosa[1]

[1] Institute of Informatics and Applications. Agents Research Lab
University of Girona. Campus Montilivi, Building P4. E-17071 Girona, Spain
{gustavog, blopez, peplluis}@eia.udg.es
[2] GREC. Knowledge Engineering Research Group. Technical University of Catalonia.
Campus Vilanova, Building VG2. E-08800 Vilanova i la Geltrú, Spain
cecilio.angulo@upc.es

Abstract. Smart User Models improve the quality of services personalization reducing the overload of processed information and capturing the affectivity of the user in the next generation of open, distributed and networked environments in Ambient Intelligence. In this paper, we combine the flexibility of intelligent agents with the information processing capabilities of Support Vector Machines in order to capture the most relevant preferences, tastes and behaviors of the user through an incremental learning process. Mainly, a multi-agent architecture is developed in order to manage services and user preferences in several domains. The set of functionalities and capabilities of each agent in the multi-agent Smart User Model is described and illustrated with a case study.

1 Introduction

The most generalized vision of Ambient Intelligence draws an open and networked world of all kinds of objects. However, the center of this world is the user that needs to satisfy his/her preferences through personalized services in this sort of open, distributed, heterogeneous and interconnected environment. The user envisioned in the Ambient Intelligence is the situational human being making decisions not only based in his/her preferences, tastes and interests, but also influenced by his/her perceptions about the context. The context is a multidimensional parameter that includes time, place, weather, emotions among others variables. Personalization in Ambient Recommender Systems can be achieved through internal representations of the users in the devices, i.e. user models. Such artificial representations have been mainly studied by the Human-Computer Interaction community for years, however, the development of applications in open environments such as Ambient Intelligence and Internet poses the challenge of modelling the user once and continuously and, what is more important, the use of a unique model for all applications with which the user interacts. In order to contribute to such kind of future user models, we combine the synergy of smart

adaptive systems, intelligent agents and Support Vector Machines to develop a Multi-agent Smart User Model (*SUM*). The proposed *SUM* is able to deal with any type of objective, subjective and emotional features, explicit or implicit, of the user in several domains and it continuously increases the knowledge of the user preferences and interests in an unobtrusive way. The flexibility (re-activity, pro-activity and social-abilities) of agents are a cornerstone to implement the *SUM* in Ambient Recommender Systems.

Ambient Recommender Systems pro-actively operate like an 'adviser' on the behalf of users. Their value added is based on suggesting suitable advices, recommendations, or predictions of interest for each user in his/her particular context. Particularly, our challenge is to develop a unique user model that influence the decision process of recommender systems in order to give a relevant advice, a recommendation or a suggestion of an item or service to the user in several domains. The figure 1 shows different geometric figures, which represent the domains (for instance, restaurants, movies, music, news) at domain level. These interact by means of wrapper agents with corresponding geometric figures which, represent different recommender systems (restaurant recommender system, movie recommender system, music recommender system, news recommender system) at the computational level. In this level we can see the Multi-agent Smart User Model which interact with the user through hand-held or desktop devices.

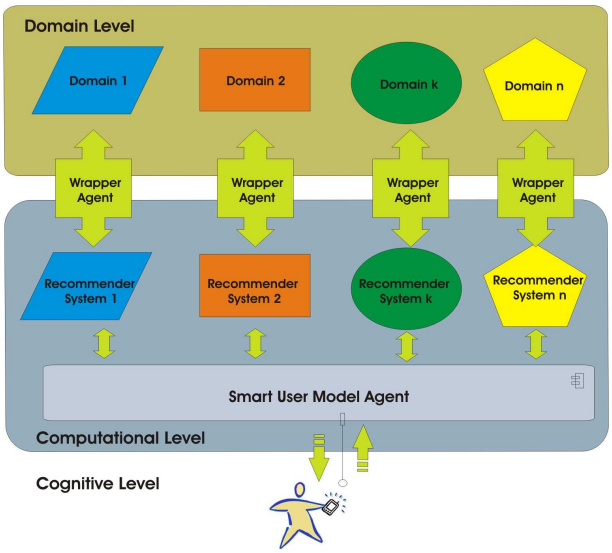

Fig. 1. The Smart User Model in Ambient Intelligence

We are concerned with the ongoing work of an unobtrusive adaptive user model. Since our approach to user modelling encompasses the communication (inter operability) and coordination (coherent actions) with recommendation

processes, agent technology provides us the appropriate flexibility to achieve all such issues. Consistently, we present a developed prototype of user model as a multi-agent system, which is able to provide services through its proactive, reactive and social capabilities to other agents, applications and users. The Smart User Model is able to provide information about the user when a new application in the environment requires it (reactivity); it is able to search new applications in which the user can be interested (pro-activity) and it can interact with other user models to obtain recommendations in a collaborative way (social-ability).

This paper is organized as follows. In section 2, we describe the Smart User Model. Section 3 is devoted to introduce the support vector machines kernel method and its relation with the *SUM*. In section 4 the multi-agent architecture is explained. We continue on section 5 with an example of use of our user model for multiple domains, and we end in section 5 with some conclusions and discussion.

2 Smart User Model

Broadly speaking, a Smart User Model should be able to deal with any type of objective, subjective and emotional features, explicit or implicit, of the user. For such purpose, in [1] has been defined the following *Smart User Model*,

$$SUM = \left\{ \begin{array}{c} \left[\left(a_1^O, v_1^O\right), \ldots, \left(a_i^O, v_i^O\right), \ldots, \left(a_n^O, v_n^O\right)\right] \; ; \\ \left[\left(a_1^S, v_1^S\right), \ldots, \left(a_j^S, v_j^S\right), \ldots, \left(a_m^S, v_m^S\right)\right] \; ; \\ \left[\left(a_1^E, v_1^E\right), \ldots, \left(a_k^E, v_k^E\right), \ldots, \left(a_l^E, v_l^E\right)\right] \end{array} \right\} = \left\{U^O; U^S; U^E\right\}$$

where the collection of attributes-value pairs, $U^F = \left[\left(a_p^F, v_p^F\right)_p\right]$ represents n objective *(F=O)*, m subjective *(F=S)* and l emotional *(F=E)* features of the user. In this form, each user behavior is captured by a Smart User Model, SUM, defining his/her internal representation in the environment, to achieve ambient-aware personalization.

In order to extend the use of the SUM in several application domains, we initially define the user model, *UMD*, for a given existing application domain i as follows,

$$UMD_i = \left\{A_i^D, A_i^I, A_i^S\right\}$$

where A^D, A^I, A^S are the sets of domain characteristics, interests and socio-demographic features, respectively, of the user required by the specific application.

Then, we establish a relationship between the general internal SUM, and the user model for a specific application domain, UMD_i, by means of a weighted graph, where $UMD_i = G\left(SUM, UMD_i\right)$. Such a graph connects user's internal features of the SUM with particular user features required at the application domain UMD_i. Particularly, emotional features of the SUM, U^E, modifies the weights used on the graph according to the emotional state of the user (For more details see [1]). The methodology for managing objective and subjective user features is based on the combination of machine learning methods: inductive methods for generalization, in particular support vector machines, and deductive

methods for specialization. This methodology can be used to both, learn user features from user information stored in recommender systems and deliver the user features to other recommender systems. For details on the SUM management, see [2].

Therefore, user's *UMD* for each application are defined by shifting information from and to *UMD*'s of different existing domains according to the weighted graphs $G\left(SUM, UMD_i\right)$ defined by each application where user interplays.

3 Support Vector Machines in User Modelling

The Support Vector Machine (SVM) is a type of learning machine for summarizing information and modelling from examples based on the statistical learning theory, which implements the structural risk minimization inductive principle in order to obtain a good generalization from data sets of limited size [3, 4]. There has been a great deal of research interest in these methods over the last years, because:

- They provide good generalization on the data.
- They are well suited for sparse data.
- They exhibit independence of the results from the input space dimension.

Although initially conceived for linearly separable two classes classification problems, new algorithms have already been derived to solve classification problems with non-separable data, regression, ordinal regression, and multi-class problems. Let $\mathcal{T} = \{(\mathbf{x_i}, y_i)\,;\, \mathbf{x_i} \in \mathcal{X}, y_i \in \{-1, +1\}\}$ be a training data set for a binary classification task, where classes are labelled as +1, -1. Let the decision function based on a hyperplane be $f(\mathbf{x}) = sign(\mathbf{w} \cdot \mathbf{x} + b)$. According to the statistical learning theory, a good generalization is achieved by maximizing the margin between the separating hyperplane, $\mathbf{w} \cdot \mathbf{x} + b = 0$, and the closest data points for each class in the input space. This optimal hyperplane can be determined by solving a quadratic programming problem. The decision function can thus be written as,

$$f\left(\mathbf{x}\right) = sign\left(\sum_{i=1}^{SV} \alpha_i y_i \left(\mathbf{x_i} \cdot \mathbf{x}\right) + b\right)$$

In order to expand the method to non-linear decision functions, the original input space, $\mathcal{X}$, projects to another higher dimension dot product space $\mathcal{F}$, called feature space, via a nonlinear map $\phi : \mathcal{X} \rightarrow \mathcal{F}$, with $dim(\mathcal{F}) >> dim(\mathcal{X})$. In this new space the optimal hyperplane is derived. Denoting the inner product in $\mathcal{F}$, (kernel) $\phi(\mathbf{x_i}) \cdot \phi(\mathbf{x_j}) = K(\mathbf{x_i}, \mathbf{x_j})$, the decision function is formulated in terms of this kernel.

$$f\left(\mathbf{x}\right) = sign\left(\sum_{i=1}^{SV} \alpha_i y_i K(\mathbf{x_i}, \mathbf{x}) + b\right)$$

As an important consequence of the SVM procedure, just a few of the training patterns are significant for classification purposes, those having a weight α_i

non-zero. These elements lie on the margin of the class and them are known as support vectors. This means that the representation of the hypothesis generated by the SVM is solely given by the points that, in the input space, are closest to the hyperplane and therefore these are the patterns most difficult to classify. The patterns that are not support vectors do not influence the position and direction of the decision function and are therefore not relevant to the hypothesis. Moreover, for this methodology the original space does not have to be an Euclidian space. By using appropriated kernels, any original space ('words', 'figures', 'strings', 'preferences', 'attributes') can be transformed with minor restrictions in an useful feature space, $\mathcal{F}$. Support Vector Machines are suitable in order to implement efficient kernel methods to process very large and high-dimensional data sets produced by Ambient Recommender Systems in several domains. Several kinds of data sources for user modelling, such as, weblogs, socio-demographic databases, transactional databases, preferences and attributes databases and sensory databases among others can be pre-processed efficiently with SVM. We have implemented a One-Class SVM like a learning component of the multi-agent system defining a ranking of user preferences.

4 Multi-agent Smart User Model Architecture

To support our *SUM* approach on a web-based application, we propose a multi-agent architecture defined at two main levels of abstraction [5]. At the highest level, two abstract agents exist (see Figure 2): the *Web Service Abstract Agent* (*WSAA*) and the *Ubiquitous Abstract Agent* (*UAA*). The *WSAA* provides capabilities of autonomy regarding the automatic discovering of services in the Internet for the user [6]. It communicates with the applications in a specific domain. When applications are non agent-based, a *wrapper agent* operates like a middleware between the *WSAA* and the application. The *UAA* gives initialization, identification, interoperability, control, coordination and management of the user preferences allowing a flexible and autonomous human-agent interaction. It is a generic and portable user model working according to our definition of *SUM*.

Coordination between *WSAA* and *UAA* is established mainly by two mechanisms: (i) *WSAA* requests to *UAA* personalized information to deal with the applications in the environment (recommender systems); (ii) *UAA* receives information from *WSAA* regarding the success or failure of the application interaction. Such relevance feedback is used by the *UAA* to learn about the user interest, so the corresponding *SUM* and the weighted graph $G(SUM, UM_i)$ of the application is updated.

Both, the *WSAA* and the *UAA* are designed to be implemented in a distributed platform. The *WSAA* can be stored in a server while the *UAA* in a mobile device. At the next abstract level, both abstract agents are implemented as multi-agent systems, as we explain in the remaining of this section.

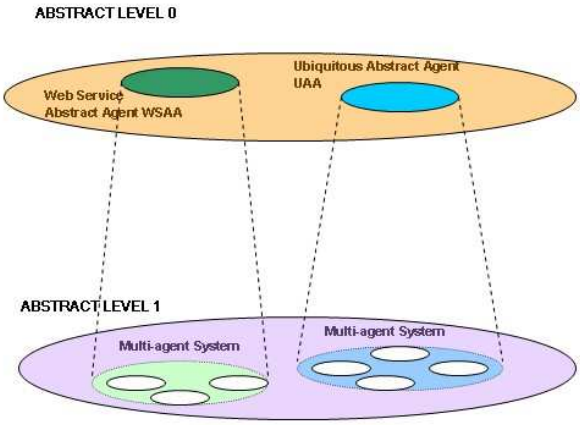

Fig. 2. Two-level Abstract Architecture for the Smart User Model

4.1 WSAA Architecture

Three types of agents compose the *WSAA*, namely (see Figure 3):

Accountant Agent. It maintains a register of user- interacted applications and domains. It also requests to the *UAA Application Agents* the establishment of new applications (see subsection 4.2).

Provider Agent. Using contextual information and interacting with the *UAA Repository Agent* it captures the pro-active behavior of the user by finding new applications in not registered domains in which it can be interested.

Consumer Agent. It finds a user requested service by communicating with the *Provider Agent*, up-loading the service and creating an *Application Agent*.

4.2 UAA Architecture

The *UAA* has four types of agents, namely (see Figure 3): *Control Agent, Creator Agent, Application Agents* and *Repository Agent* (see Figure 3).

Control Agent. Its tasks are: (i) user login service; (ii) to dialogue with the user regarding his/her interaction with an application (suggested by the *WSAA* or requested for the user); (iii) to request to the *Creator Agent* for the generation of an *Application Agent* to manage the application confirmed by the user.

Creator Agent. It is a temporal agent managing the user information in previous-applications the first time that him/her is registered in the system. It has three goals: (i) to acquire the user profile by capturing all the information spread in his/her interaction with recommender systems, and communicate

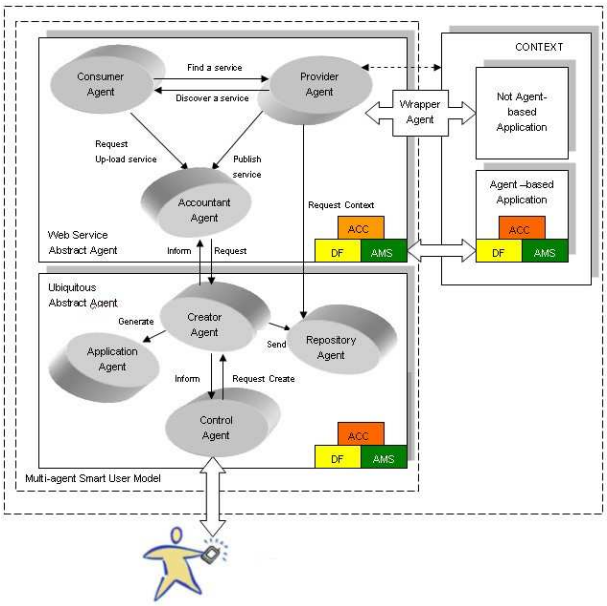

Fig. 3. Multi-agent System Architecture for the Smart User Model

it to the *Repository Agent*; objective, subjective and emotional features of the user are learned via the methodology described in [1, 2]; (ii) to generate *Application Agents* from past user interactions that will be in charge of the interaction of the user and the application from now on; (iii) register the previous applications in the multi-agent system by means of the *Control Agent*. To improve the performance of the *UAA*, once the *Creator Agent* has realized its functions, it is removed.

Application Agents. Dynamically created when interaction with an application exists, the number of *Application Agents* varies from user to user. They provide all the information about the user that an application requires (reactivity), by acquiring and saving the relationship graph between the *SUM* and the user model of the application UM_i. Endowed with social abilities, *Application Agents*, are connected with other multi-agent *SUM*'s, establishing a social network [7] of Smart User Models. When more than one agent has interacted with a certain application, so, more than one possible graph in the social network can be found, the *Application Agent* composes the graphs by means, for instance, of trust measures [8]. Else, if no agent of the social network has interacted with the application, the intervention of the user is required to establish the graph.

Repository Agent. It provides database storage procedures to save the knowledge of the user represented at the *SUM*. Individual user information is kept in a non-redundant, complete and consistent way in order to share it when and where necessary.

5 A Case Study

In this section, we illustrate with an example the functional operation of the architecture proposed. Let Juán Valdez be a user that has interacted in the past with the IRES recommender system in the 'restaurant domain' [9]. Now, Juán Valdez sets up his *SUM*, therefore the *UAA* starts.

In a first step, Juán Valdez initializes his Smart User Model through the *UAA* by registering his ID and his password through the *Control Agent*. Immediately the *Control Agent* requests to the *Creator Agent* for registration. Such latter agent, first, gathers the current information about the user in the restaurant domain, and sends it to the *Repository Agent*. Then, the *Control Agent* creates an *Application Agent* for the restaurant recommender system, and registers the restaurant application to the *Control Agent*

At the *UAA*, the *Control Agent*, prompts the user the information regarding cinema recommender systems and Juán Valdez selects one application. Then, the *Control Agent* creates an *Application Agent* to deal with the new application. The *Application Agent* looks in the social network for a user that has deal with the new application. It is the case, that Paula Allende has already interacted with the new application. So, the corresponding *Application Agent* of Paula and Juan dialog. The *Application Agent* of Juán acquires the graph $G\,(SUM, UMD_i)$ corresponding to the relationship between the *SUM* of Paula and her *UMD* in the 'cinema domain'. Weighted graph G is adapted to the *SUM* of Juán Valdez and his *Application Agent* is ready to deal with the recommendation process.

After a while, the *WSAA Provider Agent* gathers information about new recommender systems in the 'restaurant domain'. That is the case, Juán Valdez has used until now a recommender system of the Girona city, and the *Provider Agent* has discovered a recommender system about the Barcelona city. Since the user is travelling round this latter locality (contextual information), the *Provider Agent* believes that such information can be interesting for the user. Hence, the *WSAA* requests to the *UAA* about the possibility of generating a new application on this new recommender system.

6 Conclusions

In the past, user modelling had focused its attention in developing domain-dependent software architectures for user models [10–13]. However, in recent years, information technology has moved from single and centralized uses to distributed multipurpose systems, which are now increasingly embedded in a fully interconnected world [14, 15]. The Smart User Models not only must have relation with the context in where these are used but also these are a cornerstone in cross-domain recommendation processes. For instance, if a user model helps to a user to choose a comfortable restaurant, this same user model could help him/her to choose a comfortable cinema, although the domains are different. Therefore, we are contributing to the next generation of open environments through Smart User Models which include between others the emotional factor of the user, which they represent.

In this paper, we have presented a multi-agent architecture for Smart User Models that aims to shift traditional user models to this a new vision of distributed models. Our multi-agent Smart User Model can operate across multiple domains implemented by a dynamic composition of agents' services. The architecture is defined at two abstract level, one concerning services and another one dealing with user features, constituting a distributed platform.

We are currently testing our hypothesis with the use of kernel-based methods [16–18] in order to construct an automatic mapping of user features into the high-dimensional features space of several domains. The Multi-agent Smart User Model would contribute with the quality of life through a re-usable and non-intrusive intelligent adaptive system with ability to understand user habits. We think that in a near future our model will provide a rich workbench to test learning methods (acquisition and information shift of user features) in open environments.

7 Acknowledgments

This research has been supported by the Spanish Science and Education Ministry project TIN2004-06354-C02-02.

References

1. González, G., López, B., de la Rosa, J.: Managing Emotions in Smart User Models for R ecommender Systems. In: Proceedings of the 6th International Conference on Enterprise Information Systems (ICEIS'04)., Porto, Portugal (2004) 187–194
2. González, G., López, B., de la Rosa, J.: Smart User Models for Tourism: An Holistic A pproach for Personalized Tourism Services. ITT Information Technology & Tourism Journal **6** (2004) 273–286
3. Vapnik, V.N.: Statistical Learning Theory. John Wiley & Sons, New York (1998)
4. Cristianini, N., Shawe-Taylor, J.: An introduction to Support Vector Machines and other kernel- based learning methods. Cambridge University press 2000 (2000)
5. Giret, A., Botti., V.: Towards an Abstract Recursive Agent. Integrated Computer-Aided Engineering **11** (2004)
6. Dale, J., Lyell, M.: Towards an Abstract Service Architecture for Multi- Agent Systems. In: Challenges in Open Agent Systems '03 Workshop., Melbourne, Australia. (2003)
7. Palau, J., et. al.: Collaboration Analysis in Recommender Systems Using Social Networks. In Klusch, M., Ossowski, S., Kashyap, V., Unland, R., eds.: Cooperative Information Agents VIII: 8th International Workshop, CIA 2004. Volume 3191 of Lectures Notes in Computer Science., Erfurt, Germany, Springer-Verlag Heidelberg (2004) 137–151
8. Montaner, M., López, B., de la Rosa, J.L.: Opinion- based Filtering Through Trust. In Matthias Klusch, S.O., Shehory, O., eds.: Proceedings of the 6th International Workshop on Cooperative Information Agents (CIA'02). Lecture Notes in AI, Madrid (Spain), Springer-Verlag Berlin Heidelberg (2002) 164–178
9. Montaner, M., et.al.: IRES: On the Integration of Restaurant Services. AgentCities Agent Technology Competition: Special Prize, Barcelona (Spain). (2003. Available at: http://arlab.udg.es/GenialChef.pdf)

10. Fischer, G.: User Modeling in Human Computer Interaction. User Modelling and User-Adapted Interaction (UMUAI). **11** (2001) 65–86
11. Kobsa, A.: Generic User Modelling Systems. User Modelling and User-Adapted Interaction (UMUAI). **11** (2001) 49–63
12. Brusilovsky, P.: Adaptive Hypermedia. User Modelling and User-Adapted Interaction (UMUAI). **11** (2001) 87–110
13. Fink, J.: User Modelling Servers: Requirements, Design, and Evaluation. PhD thesis, Dept. of Mathematics and Computer Science, University of Essen, Germany (2003)
14. Muñoz, M.A., et.al: Context-Aware Mobile Communication in Hospitals. IEEE Computer **36** (2003) 38 – 46
15. Sadeh, N.M., et. al.: Creating an Open Agent Environment for Context-A ware M-Commerce. In et al., B., ed.: Agentcities: Challenges in Open Agent Environments. Lecture Notes in Artificial Intelligence, Agentcities, Springer Verlag (2003) 152–158
16. Angulo, C., Català, A.: Ordinal Regression with K-SVCR Machines. In Mira, Prieto, eds.: 6th International Work-Conference on Artificial and Natural Neural Networks, IWANN 2001. Granada, Spain. Volume 2084 of Lecture Notes in Computer Science. (2001)
17. Angulo, C., Parra, X., Català, A.: K-SVCR. A Multi-class Support Vector Machine. Neurocomputing. Special issue: Support Vector Machines **55** (2003) 57–77
18. Zhang, T., Iyengar, V.S.: Recommender Systems Using Linear Classifiers. Journal of Machine Learning Research **2** (2002) 313–334

Combining Stereotypes for Robust Information Prioritization

Zoë Lock[1,2] and Daniel Kudenko[2]

[1] QinetiQ, Malvern Technology Centre, St Andrew's Road, Malvern, WR14 3PS, UK
zplock@qinetiq.com
[2] Department of Computer Science, University of York, Heslington, York, YO10 5DD, UK
{kudenko, zlock}@cs.york.ac.uk

Abstract. In agile team settings, such as military command, flexible user modeling is required to respond to major shifts in user requirements triggered by changes in team membership or role assignment. Single-component user models are not robust enough for such situations. In this paper, we describe and evaluate a modular user modeling approach, more appropriate to use in agile team settings. We show that this approach performs as well as a single-component approach in terms of average precision while being more portable, to new circumstances, for many users. We also show how the new approach can address the new user problem for many users.

1 Introduction

In this paper, we describe an investigation into the robustness of a modular user modelling approach for textual information prioritization[1]. Our work is driven by the need for an effective information prioritization tool for use in agile military environments in which a user's team membership, role assignment and task allocation, all of which dictate the user's information requirements, are subject to wide-reaching and abrupt changes. We believe that flexibility is an important attribute of user models alongside mobility and pervasiveness in that the user models used to prioritize information in a personalised manner remain useful despite such changes.

Most user modeling approaches represent the attributes of an individual user within a single, monolithic structure. Such a model is brittle in response to sudden and significant changes in the circumstances of the user. In addition, this single-component approach suffers from the new user problem [1]: when a new user joins a team, there are no examples of how he rates items. The user will then be required to provide examples with which to train a new model from scratch, even though his role and team membership might already provide useful information that could serve to reduce the training overhead.

To overcome the above limitations of the monolithic approach, a modular approach is required. Rich introduced *stereotypes* as components that could be combined to

[1] This work was carried out as part of the UK Ministry of Defence Corporate Research Programme.

construct modular user models [2]. A stereotype is itself a model of the common attributes or interests of a group of users. Once a user has been tagged with a stereotype, some of his attributes can be inferred with minimal information from the user. Users can be tagged with multiple stereotypes and if these differ in their knowledge of a particular trait of a user then this is resolved automatically using the confidence the stereotypes have in their knowledge.

Our investigations focus on the robustness of modular user models for text prioritization, based on stereotypes, to changing requirements and new users. In our work, stereotypes correspond to teams and roles, which are subject to sudden changes, rather than personality traits such as *feminist* or *sporty*, which are normally more persistent and may change gradually over long periods of time.

The modular approach has the advantage that when a perspective changes, the corresponding stereotype can be replaced with another that is more appropriate to the new situation. All other stereotypes, assumed to be unaffected by the perspective change, remain in the overall user model unchanged. The updated user model can then be used to rate new items without the need to retrain a new user model from scratch. Perspective changes are discussed further in section 5.2.

Another advantage is that existing stereotypes can be used to construct an initial user model for a new user. In this way, the new user problem can be alleviated using a modular approach as long as the stereotypes of the new user are known. Our approach to the new user problem is described in more detail in section 5.3.

The paper is structured as follows. Section 2 is a short overview of relevant modular modeling approaches. Section 3 describes our approach to learning modular user models. Section 4 details the two data sets used for our investigation. Section 5 outlines the results of the experiments. Finally, the conclusions and an outlook to future work are presented in section 6.

2 Related Work

Within the field of user modeling, some modular user models have been developed in which the interests of a user is divided into multiple topics of interest (such as football or local news), each of which can be represented by a separate model component. In most cases, only one component is used at any one time and so the relevant component must be selected to suit the current query or interest [3,4]. In our approach, a set of components will all contribute towards the rating of a new item. Baudisch and Brueckner describe a system in which the outputs of multiple queries are fused, however, these queries are not shared between multiple users so do not constitute stereotypes [5]. In another TV program recommendation system, household interest models are represented within a modular system but each stereotype contributes to the same extent to the overall interest of all users [6] so the system performs poorly for household members who do not share many interests with their housemates. More details about these modular approaches can be found in [7].

More recently, group modeling approaches that capture the attributes of multiple users have been subject to investigation. Many of these are used to make classification or ranking decisions for a group rather than the individual user. In

effect, they construct stereotypes that perform well for a group of users. For example, Masthoff has developed a mechanism for constructing a TV program schedule for a group of users from their individual ratings of the programs [8].

None of the user modeling approaches use teams and roles to construct stereotypes and modular user models.

Within the field of machine learning, the use of committees or ensembles of models is an active research topic. The central motivation for this is that models formed by integrating multiple models can be more accurate than the individual component models. There are now many different approaches to learning and integrating multiple models including stacking [9] and delegation [10]. Many of these approaches either involve training models using different classification algorithms or training models of the same type with different parts of the training set. The outputs of these models are then combined or fused in some way to output a single classification or ranking decision. Rather than accuracy, our central motivation for combining models is to increase the robustness and flexibility of user models in the face of changing requirements.

3 Our Modular User Modeling Approach

This section outlines our approaches to constructing both single-component and modular user models. User models and stereotypes are automatically derived from a set of text items, binary relevance feedback on those items from users and the team and role assignments of those users using text analysis and machine learning.

Firstly, we perform feature selection in order to reduce the size of the data set and to construct initial single-component user models and stereotypes. In our text domain, features correspond to words appearing in the data set documents. In the case of a single-component user model, feedback of an individual is used to train the model. In the case of a stereotype, feedback from multiple users is used[2]. This task involves three steps:

1. standard stop words are removed from each item;
2. a statistical metric is applied to the set to score each remaining word according to its indication of relevance in a training data set[3];
3. the top n scoring words are used to seed the stereotype[4] and their scores are normalized to sum to 1.

For a fair comparison between modular and single-component user models, each type is trained to contain the same total number of features. If the number of features selected for each stereotype in a user's modular user model is n then the number of features selected for a single-component user model is n * the number of stereotypes that apply to the user. All stereotypes and single-component user models in the

[2] Handling disagreements between stereotype members in an important issue but is not be discussed here during to space constraints.

[3] We have used the χ^2 statistical metric (numerous metrics were compared in [11])

[4] Unless otherwise stated, n = 10 as preliminary experiments indicated that the value gave most consistent performance.

experiments are learnt using same technique to remove confounding systemic differences (an important issue noted in [12]).

To rate an item according to its relevance to a trained single-component user model or single stereotype, the weights of any of the *n* words contained within the model or stereotype and appearing in the item are summed together to obtain a single relevance rating between 0 and 1.

A modular user model consists of a set of stereotypes and their associated personalized *inter-component weights*. The inter-component weights (between ±1.0) indicate the relative contributions the stereotypes make to overall relevance of items to a user. The user's relevance feedback is used to train the inter-component weights: the weights are all initialized to 0 then a simple gradient descent algorithm [13] is used to train their initial values so that the inter-component weights of stereotypes that contribute most to accurate overall relevance ratings are higher than others. A new, unseen text item is rated by a modular user model by computing the linear weighted sum of the ratings of its stereotypes.

We claim that the modular models:

1. are comparable in prioritization performance to single-component user models;
2. are robust to changes in user perspectives;
3. can be used to alleviate the new user problem.

4 Evaluation Data Sets

The few publicly available relevance feedback data sets on the WWW that can be used by researchers to evaluate their information prioritization systems do not involve to team settings so they do not declare the teams and roles of users so are not directly applicable to our approach. In order to stimulate experimentation, other relevance feedback data was sought. We obtained two sets of data; one from a military source and one from a non-military source. The data sources are described below. Users were asked to rate an item as relevant if he would make use of any of the information contained within it when performing his declared roles and irrelevant otherwise.

The military data came from an experimental planning and execution task for two UK Joint Force Component Headquarters teams: Land and Air. Both teams contained five team members with the following roles: Chief-of-Staff; Intelligence officer; Plans/Operations Officer; Logistics Officer; Liaison Officer. Each role was assumed by exactly one participant and each participant belonged to one of the two teams. An ex-military officer provided a set of 133 realistic text documents for the experiment. Explicit, binary feedback on the text items was obtained from the 10 users.

A research group at QinetiQ has its own Intranet environment in which pages may be added, edited and viewed by group members. The environment is used to store and share useful information concerning research projects and other technical matters. 84 pages about diverse projects and topics were taken from the Intranet and used as information items. Group members work in several different project teams and adopt different roles within those teams. Explicit, binary relevance feedback was obtained from 13 users.

5. Results

This section details the results of our experiments to assess the three claims in section 3. The primary evaluation measure for our investigation is average uninterpolated precision or average precision (AP). This metric assesses how good a user model is at pushing relevant items to the top of the ranking above irrelevant ones (prioritization):

$$AP = \frac{1}{N}\sum_{i=1}^{N} P_i(r)$$

where $P_i(r)$ is the precision at relevant retrieved document i and N is the number of relevant documents retrieved by the ranking algorithm (true positives). A score of 1.0 means that all relevant items are ranked above all irrelevant items.

Stratified ten-fold cross validation [14] was used for all the experiments described in this paper. Paired t-tests have been used to analyze the differences between the performances of different models ($p < 0.05$ is deemed significant).

5.1 Comparison Between Single-Component and Modular User Models

The first experiment was run to assess claim 1: that modular user models can provide the same levels of performance than that of single-component ones. Table 1 and Table 2 show the AP results for the Intranet and military users for the single-component approach and the modular approach. For both data sets the performances of the trained modular models are comparable to that of the single-component models.

Table 1. Performances of single-component (SC) and modular approaches (MOD) for the military data domain

User	SC	MOD
Air – Intelligence Officer (A2)	0.97	0.98
Air – Plans/Operations Officer (A3)	0.51	0.43
Air – Logistics Officer (A4)	0.78	0.83
Air – Chief-of-Staff (AC)	0.40	0.41
Air – Liaison Officer (AL)	0.72	0.66
Land – Intelligence Officer (G2)	0.96	0.92
Land – Plans/Operations Officer (G3)	0.41	0.50
Land – Logistics Officer (G4)	0.65	0.52
Land – Chief-of-Staff (GC)	0.45	0.37
Land – Liaison Officer (GL)	0.82	0.77
Average	0.67	0.64

Table 2. Average precision values of the single-component (SC) and modular user models (MOD) approaches for the Intranet data domain

User	SC	MOD
1	0.71	0.76

2	0.75	0.73
3	0.79	0.81
4	0.81	0.82
5	0.63	0.45
6	0.80	0.79
7	0.45	0.58
8	0.92	0.97
9	0.37	0.45
10	0.74	0.79
11	0.54	0.41
12	0.58	0.54
13	0.65	0.69
Average	0.67	0.68

5.2 Robustness to Perspective Changes

In section 3, we claimed that our modular user models are robust to major shifts in a user's information requirements caused by changes to team membership or role assignment. The advantage of this feature is a more rapid response to a new situation as the need for training is alleviated. In this section, we present results to empirically support the claim. Ideally, to test the robustness of modular user models, relevance feedback should be collected from user experiments in which users do indeed change team or role whilst rating items. The military and Intranet experiments did not involve such changes so we have used the existing data to simulate the situation in which information requirements change abruptly.

Consider two users, U1 and U2, where U1's model contains stereotypes C1, C2 and C3 and U2's model contains stereotypes C1, C2 and C4. U1 and U2 have stereotypes C1 and C2 in common but each has one stereotype the other does not (C3 and C4). If C3 and C4 represent team stereotypes and U1 and U2 swap teams, then the new model for U1 should contain C1, C2 and C4 stereotypes and the new model for U2 should contain C1, C2 and C3 – making U1's new model the same as U2's old model and vice versa. If it is assumed that a user's requirements depends only on his team and role assignment (which is reasonable given the rating criteria used by the users given in section 4), then after stereotype swapping, U1's new requirements are now equivalent to U2's old requirements and vice versa. Based on this assumption, U1's new model can be tested on U2's test feedback, which was obtained before team swapping, and vice versa. If U1's new model performs well against U2's feedback then the modular user model is robust to changes in perspective under the assumption made.

In general, for each pair of users who share all but two stereotypes of the same type (role or team), the two stereotypes that are not shared are swapped (keeping the inter-component weights static) to create two new user models. The results of the robustness experiments are given in Table 3 and Table 4. Each pair of users who swap stereotypes over is represented as A→B where B's test feedback is used to evaluate the performance of A's new user model.

Table 3. Average precision values for a trained single-component (SC) user model and a trained modular user model (MOD) with stereotype swapped (military)

User	AP		User	AP		User	AP	
	SC	MOD		SC	MOD		SC	MOD
A2→A3	0.32	0.48	AL→A3	0.32	0.55	GC→GL	0.77	**0.83**
A3→A2	0.97	**0.89**	A4→AC	0.49	0.52	GL→GC	0.73	**0.73**
A2→A4	0.68	**0.75**	AC→A4	0.68	**0.78**	G2→G3	0.44	0.44
A4→A2	0.97	**0.98**	A4→G4	0.52	0.51	G3→G2	0.96	**0.83**
A2→AC	0.49	0.52	G4→A4	0.68	**0.78**	G2→G4	0.52	0.49
AC→A2	0.97	**0.98**	A4→AL	0.50	0.37	G4→G2	0.96	**0.89**
A2→G2	0.96	**0.91**	AL→A4	0.68	**0.82**	G2→GL	0.77	**0.84**
G2→A2	0.97	**0.97**	AC→LC	0.73	**0.71**	GL→G2	0.96	**0.89**
A2→AL	0.50	0.37	GC→AC	0.49	0.53	G3→G4	0.52	0.46
AL→A2	0.97	**0.92**	AC→AL	0.50	0.39	G4→G3	0.44	0.43
A3→A4	0.68	**0.78**	AL→AC	0.49	0.56	G3→GL	0.77	**0.82**
A4→A3	0.32	0.46	GC→G2	0.96	**0.89**	GL→G3	0.54	0.43
A3→AC	0.49	0.53	G2→GC	0.73	**0.71**	G4→GL	0.77	**0.83**
AC→A3	0.32	0.48	GC→G3	0.44	0.44	GL→G4	0.52	0.49
A3→G3	0.44	0.44	G3→GC	0.73	**0.75**	AL→GL	0.77	**0.83**
G3→A3	0.32	0.55	GC→G4	0.52	0.49	GL→AL	0.50	0.39
A3→AL	0.50	0.39	G4→GC	0.73	**0.73**	Average	0.64	0.65

Table 4. Average precision values for a trained single-component (SC) user model and a trained modular user model (MOD) withstereotype swapped (Intranet)

User pair	AP	
	SC	MOD
2→13	0.69	0.73
13→2	0.71	0.71
3→8	0.9	0.92
8→3	0.77	0.83
6→8	0.9	0.92
8→6	0.84	0.77
7→9	0.4	0.39
9→7	0.55	0.7
11→12	0.61	0.73
12→11	0.52	0.39
Average	0.69	0.71

In both domains, there is no significant difference between the performance of A's new modular model and B's single-component model. Broadly speaking, poor performance of the modular model is matched by poor performance of the single-component model, indicating that the user's information requirements are difficult to characterize by training from his feedback. For 8 out of 10 Intranet user pairs and 25 out of 50 military user pairs, the modular user model performed well ($AP > 0.7$). These results suggest that the proposed modular user modelling approach is indeed

robust to changes in user requirements as long as the user requirements can be characterized adequately by training. When a user's circumstances do change abruptly, the modular approach can be used to generate a new user model without the need for retraining, as long as his requirements can be characterized.

5.3 Rating Items for New Users Using Modular User Models

In section 3, we claimed that the proposed modular approach would help to alleviate the new user problem. The advantage of this would be that training would not be necessary for a new user before incoming information is prioritized according to his needs. In this section, we present results to empirically support this claim.

For a new user, it is assumed that no relevance feedback has been collected but that the stereotypes to which he belongs have been declared. A single-component user model cannot be constructed in the absence of training data. On-line experiments during new users arrive and provide feedback on items presented to them, the situation is simulated using the data already obtained. In each experiment, each user in turn is selected as the new user. The new user's relevance feedback is removed from the training set so the stereotypes to which he belongs are trained without it. Given no training feedback from the new user, the inter-component weights for his model cannot be learnt in the way described in section 3. Instead, three different weight assignment approaches are used:

1. All uniform - equal weights for all the new user's stereotypes (summing to 1.0)
2. Team uniform - All role stereotypes are allocated 0.0 weighting and the team stereotypes receive uniform weighting (summing to 1.0)
3. Role uniform - All team stereotypes are allocated 0.0 weighting and the role stereotypes receive uniform weighting (summing to 1.0)

The test performances of new user modular models constructed using the approach described above are given in Table 5 and Table 6.

For 6 out of 10 military users and 8 out of 13 Intranet users, at least one of the approaches provides good prioritization performance ($AP > 0.7$). In the military domain, there are no significant differences between the performances of the three approaches. Whereas, in the Intranet domain, the team uniform approach is significantly better than the other two approaches. The reason for this may be that there are more teams than roles in the Intranet domain and some users belong to more than one team. This means that teams are more likely than roles on average to be relevance indicators. With no feedback from the new user, it would be difficult to choose the best weighting approach automatically. Given the results above, the team uniform approach could be the default choice.

Table 5. Performance of modeling approaches for rating items for a new user

User	Inter-component weighting scheme		
	All uniform	Team uniform	Role uniform
A2	0.98	0.72	1.00
A3	0.30	0.53	0.23
A4	0.83	0.58	0.84

AC	0.47	0.73	0.44
GC	0.61	0.84	0.61
G2	0.89	0.70	0.96
G3	0.38	0.47	0.22
G4	0.31	0.16	0.40
AL	0.20	0.36	0.17
GL	0.64	0.72	0.47
Average	0.56	0.58	0.53

Table 6. Performance of modeling approaches for rating items for a new user

User	Inter-component weighting scheme		
	All uniform	Team uniform	Role uniform
1	0.75	0.75	0.61
2	0.65	0.62	0.64
3	0.86	0.84	0.98
4	0.70	0.77	0.65
5	0.36	0.43	0.43
6	0.80	0.89	0.52
7	0.59	0.71	0.32
8	0.94	0.98	0.80
9	0.29	0.44	0.19
10	0.74	0.66	0.80
11	0.29	0.44	0.03
12	0.42	0.65	0.13
13	0.61	0.83	0.44
Average	0.61	0.69	0.50

6. Conclusions and Outlook

We presented our investigation into flexible modular user modeling based on teams and roles. The experimental results support the claims made in section 3. Specifically, we have shown that the proposed modular approach has comparable accuracy to a single-component approach, but has the advantage of adjusting to changes in perspective more quickly. We have shown results that suggest that stereotype swapping could be used to bootstrap user models for new users without the need for model retraining. Our future work on flexible modular user models will investigate a number of issues:

- We have seen broadly similar results when a kNN approach was used to train the stereotypes. We plan to apply the Naïve Bayes algorithm, popular in text categorization, to determine whether our results generalize to other stereotype learning methods. We also plan to generate large artificial data sets for testing.
- Diversity and accuracy are widely cited as vital factors for achieving better performance with integrated multiple models than with individual models [12,15]. However, this applies to integrated models whose constituent models

are full classifiers of a target concept. We will assess the diversity and accuracy of the team and role stereotypes to determine any effect on model performance.

- We plan to determine whether new stereotypes can be successfully added to modular user models on-the-fly in order to adapt to new circumstances.

References

1. Salton, G. & McGill, M.: Introduction to Modern Information Retrieval. McGraw Hill. (1983).
2. Rich, E. User Modeling via Stereotypes, Cognitive Science, 3(4), 1979, 329-354. (1979)
3. Billsus, D. & Pazzani, M.: A Hybrid User Model for News Story Classification. UM'99 (1999).
4. McGowan, J., Kushmerick, N., & Smyth, B.: Who Do You Want to Be Today? Web Personae for Personalised Information Access. In Proceedings of the 2nd International Conference on Adaptive Hypermedia and Adaptive Web Based Systems (2002), May 29-31, Malaga, Spain.
5. Baudisch, P. & Brueckner, L.: TV Scout: Lowering the entry barrier to personalized TV program recommendation In Proceedings of the 2nd International Conference on Adaptive Hypermedia and Adaptive Web Based Systems (2002), May 29-31, Malaga, Spain..
6. Buczak, A., Zimmerman, J. & Kurapati, K.: Personalization: Improving Ease-of-Use, Trust and Accuracy of a TV Show Recommender. TV'02:Workshop on Personalization in TV. (2002).
7. Lock, Z & Kudenko, D.: Multi-Component User Models for Personalised Briefing Agents. Workshop on User and Group Models for Web-based Collaborative Environments. Held at the 9th International Conference on User Modeling UM'03. (2003).
8. Masthoff, J.: Selecting news to suit a group of criteria: An exploration. Proceedings of the Fourth Personalized TV workshop, associated with AH04. (2004).
9. Wolpert, D.: Stacked generalization. *Neural Networks*, 5(2):241–260, (1992).
10. Ferri, C., Flach, P & Hernández-Orallo, J.: Delegating classifiers. In Proceedings of the Twenty-First International Conference on Machine Learning ICML'04. ACM. (2004).
11. Yang, Y. & Pedersen, J. A Comparative Study on Feature Selection in Text Categorization. Proceedings of ICML-97, 14th International Conference on Machine Learning. (1997).
12. Beitzel, S., Jensen, E., Chowdhury, A., Friedner, O., Grossman, D. & Goharian, N. Disproving the Fusion Hypothesis: An Analysis of Data Fusion via Effective Information Retrieval Strategies. In SAC 2003. (2003).
13. Mitchell, T.: Machine Learning. McGraw-Hill. (1997).
14. Witten, I. & Frank, E.: Data Mining. Morgan Kaufmann. (2000).
15. Kuncheva, L.: Diversity in multiple classifier systems. In Information Fusion, 6, 3-4. Elsevier. (2005).

A Specification for Agent-Based Distributed User Modelling in Ubiquitous Computing

Andreas Lorenz

Fraunhofer Institute for Applied Information Technology
Schloss Birlinghoven
53754 Sankt Augustin, Germany
Andreas.Lorenz@fit.fraunhofer.de

Abstract. This paper introduces an approach for applying agent technology for user modelling in ubiquitous computing. It illustrates the research issues in distributing the knowledge about the user across active entities and distributed user-model acquisition and application methods, and specifies the agents using a defined communication framework for distributed user-modelling for ubiquitous computing. Regarding the requirements in ubiquitous computing, co-operating agents build ad-hoc networks for receiving information from other entities and distributing knowledge to other components in the network. Therefore, the specified agents are able to react both to their environment and to messages received from neighbouring components.

Introduction

In the classic approach for *personalized system development*, the application contains specific user-modelling components inside the personalization engine. By user model acquisition, information about the user is extracted from sensoring the environment and knowledge from explicit and implicit user feedback is inferred [1]. The inferred knowledge usually is written to an internal database, mapping user attributes to their values. In the next step, both the component listening to sensor data and the knowledge-base about the user are separated from the internal application logic. In the first case, *sensor-servers* retrieve data streams from different sensors placed in the environment and deliver the information to the application. Using remote sensor-servers distributing sensor data on a network, different applications can concurrently receive the same data. In the second case, *User-Model Servers* [2] work as an application-external knowledge-base. The derived knowledge about the user is delivered to the server that hosts the information for different applications. For mobile applications, this enables systems on small devices even with limited memory and computing power to have access to meaningful user models. Furthermore, it enables different applications to have access to the same knowledge and to adapt consistently.

In the vision of ubiquitous computing the user has *one personal information space* independent of devices and the system manages the information spaces of its users.

For future application development in ubiquitous computing we expect centralized design-approaches to be confronted with uncountable clients on heterogeneous devices with different properties. In our vision, *distributed user modelling* approaches need to replace monolithic centralized user modelling by distributed user model fragments [3]. To become true, this vision requires several pre-conditions to be fulfilled:

1. The network of distributed components needs to be self-adapting, especially regarding available communication partners and technology.
2. The information needs to migrate between different hosts and platforms without being central-controlled.
3. The communication infrastructure and technical details need to be hidden from the user modelling components, and their developers.
4. Typically, application designers building distributed applications have to guarantee the following non-functional requirements: scalability, openness, heterogeneity, fault-tolerance, and resource sharing.

Facilitating communication and coordination of distributed components, we will implement cooperating agents as active components hosting on the devices and using a defined communication framework. In contrast to other approaches for applying agent-technology, sets of agents will be implemented for distributed user-modelling, user-model acquisition and user-model-application instead of a one-to-one relationship between the user and a User-Agent. Each local component might detect a section of the global state, but the network of agents will piece together these partial states for *distributed representation of knowledge about the user*. This paper demonstrates our specification of such agents.

Agent-Technology in User-Modelling

Recent *agent-based user modelling* approaches usually consists of two parts: a User Modelling Service and a User Agent (often also referred to as Personal Agent). The former keeps track of the user's interaction with the application and within the environment, and stores the inferred user and environmental characteristics. The latter usually represents the user in the system. For mobile / nomadic users, mobile agents can move with the user between devices and applications. In this section we describe recent attempts of combining user modelling and agent technologies for the application fields mentioned in the Introduction to this paper. To get an overview of the variety of agent definitions, modelling techniques, and architectures in this field, the reader is referred to [4].

Driven by the boom of web-applications in the late 1990s, the value of personalization was increasingly recognized in the field of intelligent information access on the WWW. Pazzani and Billsus [5] have introduced adaptive web site agents that recommend relevant documents to the user in an Amazon.com-like manner. They argued that the information is best used to change the behaviour of an animated agent (avatar) to assist the user. In Billsus and Pazzani [6] an intelligent information agent is considered to be a personal assistant that gradually learns about users' interests. Like the adaptive web agents presented in Menczer and Belew [7],

agent technology is either used for personalized information acquisition or for individual information presentation.

In the domain of eLearning, Vassileva *et al.* [8] base the adaptation within the I-Help system [9] on models of human users maintained by personal agents: "*Each personal agent manages a user model containing information about the user's goals (help requests, current goal), about* knowledge resources / competencies *on certain topics or tasks, and about the* relationships *existing between the user and other users.*" The Baghera project [10] has implemented personal interface agents for students and teachers, and tutor agents that base whose didactical decisions on a student model. In order to integrate human-like intelligent tutors into collaborative learning environments, Goodman *et al.* [11] have also proposed to integrate tutoring agents. These approaches have in common that student and tutor agents are connected with external user models.

Furthermore, agent technologies have been applied for personalizing location-based services like city- and tourism-guides. The Deep Map Agents introduced in Fink and Kobsa [12] provide tour recommendations, analyse spoken text, generate speech output etc. These agents, which loosely adhere to the FIPA agent specification [13], communicate to a User Modelling Server (UMS) about the user's interaction with the system and query the UMS for user characteristics. In the EU-founded CRUMPET project [14], FIPA compliant user agents are hosted on the end user terminal devices and provide the user with the service GUI. These agents adapt the information presentation to the platform evaluating the usage profile of the user.

In summary of these approaches, the agents usually query an external user model. In terms of multi-agent system development, the internal knowledge-base of such an agent actually is or refers to a user model; in terms of the general scheme of an adaptive system [1], team working agents improve user model acquisition resp. user model application.

Agent-based distributed User-Modelling

To be able to fulfil the requirements of ubiquitous computing, we propose to have a network of small active entities on the client side. Resent research in smart *sensor-networks* enables for placing huge numbers of intelligent senor-components ("smart dust") in the environment. Smart sensors are equipped with small processors that enable for intelligent information acquisition [15]. In *self-organizing networks*, such as Intel's iMote approach [16], sensor technologies build ad-hoc sensor-networks and deliver requested information on demand. Similar procedures can be applied on higher layers in the system-design. For example, modelling components receive sensor-data and distribute inferred knowledge in something like a "modelling-network", which will have effect on controlling components and so forth. In difference to sensor-networks, the components actually receive pre-processed data from virtual components instead of direct measuring the physical environment. Therefore, we propose to have *distributed active entities receiving data from and delivering information to other entities*. As active entities, software-agents have their own thread of control making them appear like "active objects with initiative" [17]

localizing not only code and state but their invocation as well. In other words, when and how an agent acts is determined by the agent. Regarding the assumption that in ubiquitous computing there won't be a central server hosting databases for all entities, the knowledge-base and the decision-finding process will be distributed across the agents. There will exist neither central user modelling nor information-acquisition/knowledge-application components.

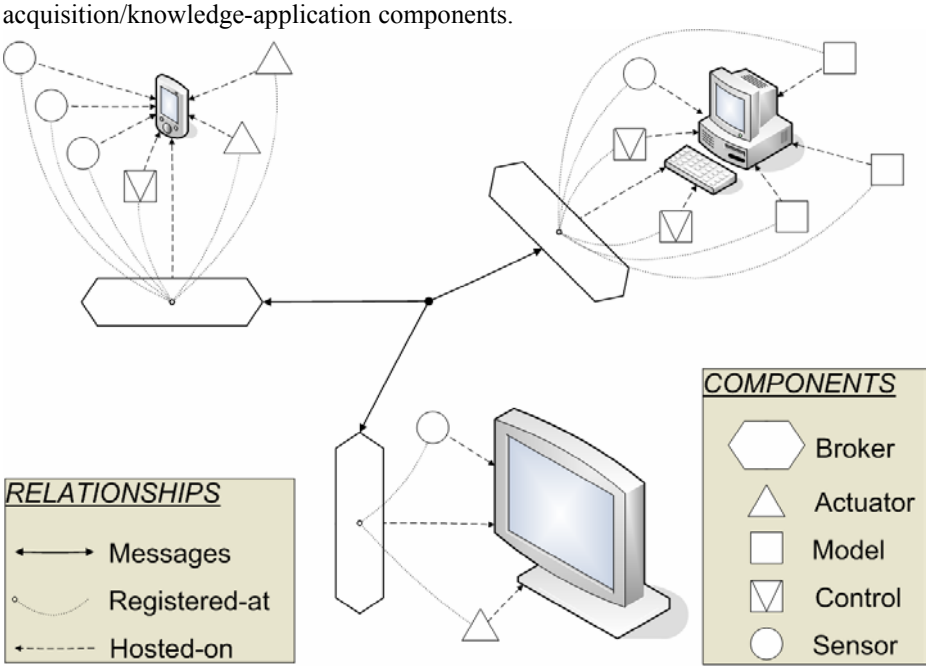

Fig. 1 Distributed User Modelling Platform

Due to the distribution of functionality and knowledge, the agents will be categorized virtually. This ensures encapsulated inter-package communication inside and broadcasting to a specific category. Though system developers are able to integrate their own packages, we propose to have four categories of *sensoring*, *modelling*, *controlling* and *actuating agent*s [18, 19, 20]. For each category, networks of highly specialized software-agents process small tasks like delivering one information snippet or deciding to display data on a particular device. Each category will be distributed over different devices, e.g. among others the light sensor of a PDA, the infrared sensor of an automatic door and the GPS-sensor of the car are part of the sensor-package regardless to their physical location and environment. In turn, each device potentially hosts agents of several categories, e.g. a PDA independently hosts sensors for light-conditions, background-noise and pen-input as well as controlling agents for content-selection and actuators for video-streaming or adjustment of the display-brightness. Fig. 1 illustrates the distributed agents hosted on different devices and their relations to each other.

Example
To illustrate the information flow between the distributed components we will shortly describe one of the application scenarios. In this scenario, the hosts illustrated in Fig. 1 are a kiosk-system and a public information display in an airport and the personal device of the traveller. The kiosk and the display are connected via LAN and the user's PDA can establish Bluetooth-connections to the kiosk-system, which is able to read the RFID-Tag fixed to the flight-ticket of the customer. The service offered to the customer is time- and route planning on the large airport: When a traveller passed by the kiosk, the public display shows the flight-number and destination, and guides the traveller to the gate anonymously, including the estimated needed time. If the customer wants to have a personal plan, she can accept the Bluetooth-connection between her PDA and the kiosk.

The communication platform

The basic underlying cooperation-approach between the agents is *cooperation by information-exchange*. Like a middleware, brokers hide the complexity of communication from the other agents. This concept can be seen in between of the blackboard-approach and the message-sending approach well-known in the field of multi-agent systems [21]: For local agents, the broker provides access to a message-board whereas the information exchange between devices is based on message-sending between the brokers (Fig. 1). The agents register at the board based on defined check-in/check-out mechanisms, announcing what information they provide and what information they request.

In the example, the brokers of the kiosk, the information-display and a database-server are connected by Ethernet continuously. The sensoring-agent, logged in to the kiosk-broker, fires the event that an identifier has been received from the customer. A controlling-agent on the kiosk listens to the event and sends out a request for the number and destination of that flight. The broker sends the request to all known brokers in the network, which is answered by an agent on the database-server. The answer – broadcasted between the brokers – is received by a listening controlling agent on the information display, who generates the command to display the corresponding data for the rendering agents on the large screen display. The information has then migrated between the distributed components with different capabilities.

So far, the knowledge about the traveller's personal data is very limited. For privacy reasons, the display will not show any private information of the customer. If desired, the traveller can accept the Bluetooth connection at the kiosk. After the connection has been established, the broker on the kiosk covers roaming between different communication-technologies: Messages received from the Ethernet-connections are also forwarded via Bluetooth to the broker at the PDA. At a glance, the network of reachable agents is extended to cover the modelling-agents of the personal attributes, goals and task of the traveller. If the traveller passes by the kiosk system, which receives and sends out the RFID-identifier on the network, the agents on the PDA also receive the information from their local broker. Local agents can

now request the position of the kiosk in order to update a local map, or controlling agents can re-arrange the time-schedule in order to skip the visit of the book-store because of potential time pressure. In turn, controlling agents can even generate commands for rendering agents on the kiosk, e.g. to display a map of nearby restaurants because of the traveller's habitat to have a coffee before boarding.

The advantage of the platform is that by standardized communication with local components distributed agents are released from discovering communication partners using different technologies. As the broker establish / loose connections to other brokers, the user model structure changes automatically and the sets of accessible information-sources and -destinations adapt to the current environment. The division of agents in different categories additionally distributes similar functionality on different devices. Regarding privacy concerns, the user has control on providing private information by allowing / disabling connections of private devices with other ones. If the connection was enabled, the global accessible user model is extended to include the attributes from personal devices at once.

Specification

Currently, our main objective is to provide a well-defined conceptual basis, in particular specifying the architecture and agents, communication and information-exchange, and cooperation-techniques and conflict-management [22], e.g. if many agents are potentially able to process the same information or agents receive ambiguous answers to a request. The realization phase has already been launched starting with the implementation of the framework and the specified communication-protocols for

- check-in/check-out
- subscribe-inform mechanism
- question-reply mechanism
- command-delivery
- acknowledgement
- exceptions

The messages are defined in EBNF and sending / receiving of such messages was implemented in several projects for receiving data from distributed sensors. In the next steps, we will finish the work on the specification, continue to implement the framework and focus on the implementation of the specified agents.

Agent-Specification

In general, we need two simple types of agents: *Information delivering agents* and *information receiving agents*, which include intelligent processing of the received information. Fig. 2 illustrates the derivation of the agents from those two basic types.

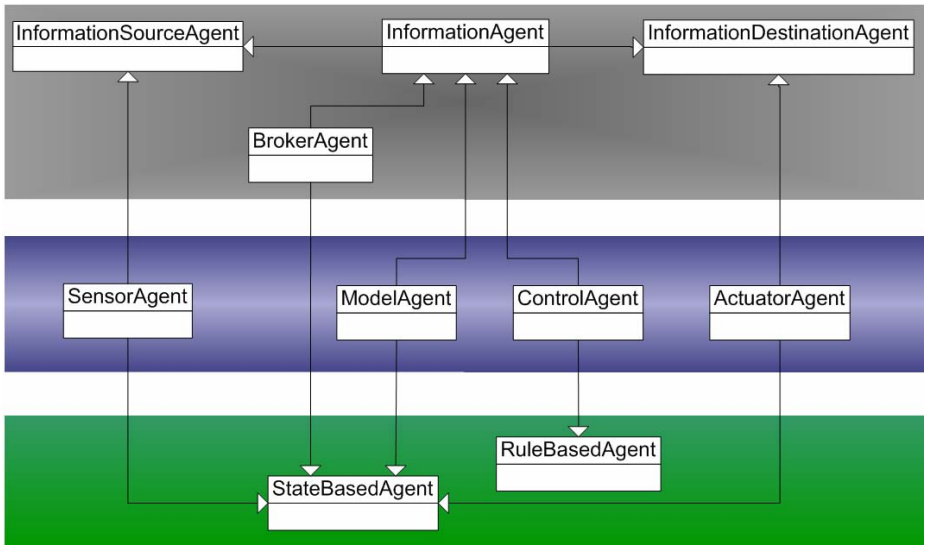

Fig. 2 The Agent's inheritance diagram

On top of Fig. 2, we have information sources on the left and information destinations on the right. Information sources do only deliver information towards other components; therefore they contain a list of all attributes they provide. The delivery can be performed either by throwing events or by answering requests from other components for specific information. In contrast, the information destinations are only able to receive information, either by listening to events or by pro-actively requesting data from others. They have an internal list of information they demand and are able to register as listeners. Derived from those two basic types, the third agent-type, the *information agent*, is able to send information as well as to receive and process information from others. As a special type of an information agent, the *broker agent* only forwards incoming information either towards other local agents or towards other brokers.

On the bottom of the figure, we depict two kinds of agent-specification from the field of multi-agent systems. Generally, we decided to model all agents in a state-based manner, except the controlling-agents. Incoming messages will trigger transitions in state-based agent-modelling, which sufficiently supports reactive behaviour and is also understandable for human developers in future. Fig. 3 exemplifies the states-diagram for the information source agents. The transitions between the states are usually defined in a transition-table.

Unfortunately, state-based agent-modelling is not applicable for knowledge-based agents. Beside the complexity of a state-diagram with many states for complex decision finding, the knowledge representation is implicit coded by the developer in the states and transitions. Adhering to state-based modelling, each derived agent would have its own state-diagram with specific conditional transitions. To determine the overall system's behaviour, a *rule-based approach* for controlling agents seems to be more appropriate. In this approach, the behaviour is coded by sets of rules of *IF condition THEN action*. Incoming messages trigger the interpretation of the rule's

conditions and fire all rules with fulfilled pre-conditions. In conclusion, we have four different models for the agents taking part in the user-modelling process:

1. Sensory agents as state-based information source agents,
2. Modelling agents as state-based information agents,
3. Controlling agents as rule-based information agents and
4. Actuating agents as state-based information destination agents.

In the next subsection we will exemplary describe the state-diagram of an information source agent in more detail.

Information source agent

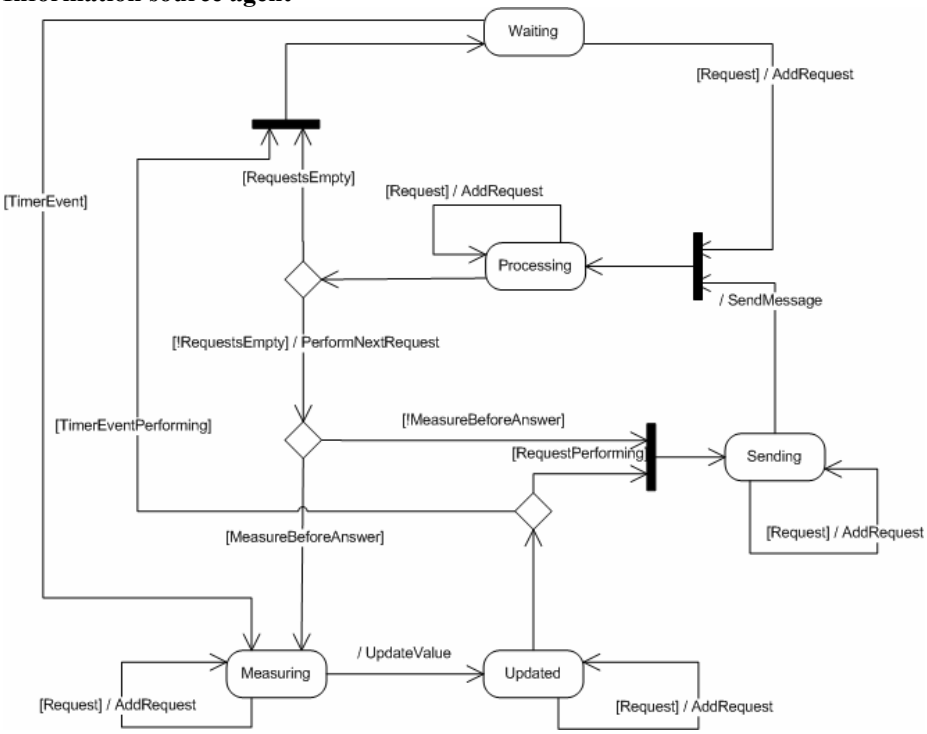

Fig. 3 The States of an Information Source Agent

The information source agent is connected either with an environmental (physical) sensor or another information source agent. The goal of the agent is to observe the parameter and to inform other agents about changes of the value. As shown in Fig. 3, an information source agent waits for external requests from other components or for timer events triggering the measurement of the observed parameter. In the former case, the incoming request is added to a list of requests to be processed, and as long as the list is not empty, the agent delivers its contained knowledge. If the agent was triggered by a timer event, or the agent is configured to measure the current value

before answering a request, the agent updates its internal knowledge-base by obtaining the current value. If the agent measures the parameter directly, it reads the physical sensor value. If it requests the information from other entities, the "Measuring"-state branches out to an internal "Requesting"-state, the agent sends the requests and waits for the answer. When the agent realises that the value has changed, it fires an event that will be delivered to all agents registered as listeners to this attribute.

Conclusions and Future Work

In this paper we have illustrated our approach of applying agent technology for user modelling in ubiquitous computing. In contrast to resent approaches, we broke the one-to-one relationship between the user and her representing User-Agent. As true for sensor-networks we choose to have many small entities cooperating in ad-hoc networks on the different devices of a user. This allows for a flexible representation of the user by assembling the knowledge of all agents reachable in the current context. For releasing the need for a mobile-agent platform, we aim at information migrating between devices instead of Mobile Agents physically moving to an unknown platform.

In the current state of platform-specification and agent-modelling, the platform and the messages being sent between the agents are implemented in several projects of our institute. In the next steps, we will focus on the implementation of the agents based on the specification presented here.

References

[1] A. Jameson. Systems that adapt to their users: An integrative overview. In *Tutorial presented at 9th International Conference on User Modelling*, Johnstown, PA, USA, 2003.

[2] J. Fink and A. Kobsa. A review and analysis of commercial user modelling servers for personalization on the world wide web. *User Modelling and User-Adapted Interaction*, 10(2-3):209–249, 2000.

[3] J. Vassileva. Distributed User Modelling for Universal Information Access. In Stephanidis C. (ed.), *Universal Access in Human - Computer Interaction*, Lawrence Erlbaum: Mahwah, N.J., 122-126, 2001.

[4] M.J. Wooldridge. *Intelligent Agents II.* Springer-Verlag, 1996.

[5] M.J. Pazzani and D. Billsus. Adaptive web site agents. *Autonomous Agents and Multi-Agent Systems*, 5(2):205–218, 2002.

[6] D. Billsus and M.J. Pazzani. User modelling for adaptive news access. *User Modelling and User-Adapted Interaction*, 10(2):147–180, 2000.

[7] F. Menczer and R.K. Belew. Adaptive retrieval agents: Internalizing local context and scaling up to the web. *Machine Learning*, 39(2/3):203–242, 2000.

[8] J.I. Vassileva, J.E. Greer, and G.I. McCalla. Openness and disclosure in multi-agent learner models. In *Workshop on Open, Interactive, and Other Overt Approaches to Learner Modelling*, Le Mans, France, 1999.

[9] J. Vassileva, G. McCalla, and J. Greer. Multi-Agent Multi-User Modelling. *User Modelling and User Adapted Interaction*, 13(1):1-31.

[10] C. Webber, L. Bergia, S. Pesty, and N. Balacheff. The baghera project: a multi-agent architecture for human learning. In J.I. Vassileva, ed., *Workshop on Multi-Agent Architectures for Distributed Learning Environments*, pp. 12–17, San Antonio, USA, 2001.
[11] B. Goodman, J. Hitzeman, F. Linton, and H. Ross. Towards intelligent agents for collaborative learning: Recognizing the roles of dialogue participant. In P. Brusilovsky, A. Corbett, and F. de Rosis, editors, *9th International Conference on User Modelling*, Johnstown, PA, USA, 2003.
[12] J. Fink and A. Kobsa. User modelling for personalized city tours. *Artificial Intelligence Review*, 18(1):33–74, 2002.
[13] FIPA Foundation for Intelligent Physical Agents. http://www.fipa.org.
[14] S. Poslad, H. Laamanen, R. Malaka, A. Nick, P. Buckle, and A. Zipf. CRUMPET: Creation of user-friendly mobile services personalised for tourism. In *2nd International Conference on 3G Mobile Communication Technologies*, pages 26–29, London, UK, 2001.
[15] M. Satyanarayanan, Of Smart Dust and Brilliant Rocks. *Pervasive Computing*, 2(4):2-4, 2003.
[16] D.E. Culler and H. Mulder. Smart Sensors to Network the World. *Scientific American*, 290(6):52-59, 2004.
[17] J. Odell. Objects and agents compared. *Journal of Object Technology*, 1(1):41–53, 2002.
[18] A. Lorenz and J. Denzinger. Functional agent systems for user modelling. *Proceedings of the Workshop on Artificial Intelligence, Information Access, and Mobile Computing*, 82–86, Acapulco, Mexico, 2003.
[19] A. Lorenz. Towards a new role of agent technology in user modelling. *Proceedings of the Workshop on Adaptivity and User Modelling in Interactive Systems*, Karlsruhe, Germany, 2003.
[20] A. Zimmermann, A. Lorenz, and M. Specht. User modelling in adaptive audio-augmented museum environments. In P. Brusilowsky, A. Corbett, and F. de Rosis (eds), *Proceedings of the 9th international conference on user modelling*, pp. 403–407, Johnstown, USA, Springer-Verlag, 2003.
[21] N.R. Jennings and M.J. Wooldridge. *Agent Technology*. Springer, 1998.
[22] C. Tessier, L. Chaudron, and H.-J. Müller. Conflicting Agents: Conflict management in multi-agent systems. Kluwer Academic Publishers , 2000.

Modeling the Effects of Task-external Status in Small, Task- Oriented, Groups

Roy Wilson*

University of Pittsburgh
rwilson@pitt.edu

Abstract. Several papers and one invited talk at UM2003 suggest a growing interest in both user and group models grounded in social-psychological research. This paper describes a simulation based on the application of a mathematically formulated social-psychological model to small, task-oriented, groups. Also presented is a simple model of influence that captures the phenomenon of "the rich get richer, the poor get poorer". These models are applied to a set of data collected independently, and for a different purpose, by other researchers. The probability that the observed differences across two experimental conditions involving a task-external status manipulation were due to chance alone is 0.138, suggesting a degree of systematicity to the status effect. The possibility that such a model could be deployed in purpose-driven decentralized group modeling is briefly considered.

1 Introduction

It has been suggested that, in building socially aware agents, it may be helpful to model status effects. Recent work has modeled: social collusion [2], and; the formation of social relationships in peer-to-peer networks [1]. Although the importance of status may be obvious, what is less obvious is how to model the interaction of task-external status and task-internal behavior. If it is important that the outcome of group discussions or deliberations be free of the influence of task-external status, it may be useful to have a theory-based model that tracks the emergence of social order within groups.

2 The Models

2.1 Status and Task Participation

The status and task participation model ($STPM$) is based on the body of social psychological research associated with expectation states theory (EST).

> [EST] ... holds that actors' behavior toward others depends on the performance expectations they hold for themselves and for others [and that] ... expectations refer to unobservable states of relational orientation to ... others. [7]

* Many thanks are due the reviewers for their helpful comments.

EST has generated a number of specific, experimentally verified, behavioral predictions that have been applied to educational settings. The $STPM$ gives an account of how stable social orders emerge within task- oriented groups, shaping the distribution of opportunities for group members to contribute to, and influencing the outcome(s) of, group discussion (or, in this paper, group decision).

The $STPM$ describes a cycle. At time T, expectation states enable and constrain the task-internal behavior of members. When actor a addresses actor b, this (probabilistically) induces a social relation R between a and b, denoted aRb. The relation R at time T constitutes a social network. The social network at time T determines the expectation states of the group members at time $T+1$. The cycle then begins again (with possibly updated expectation states).

The $STPM$ is based on a six axioms involving three parameters: π, the probability that the act of a addressing b generates aRb; η, the probability that a task-external status difference between a and b generates aRb; θ, the probability that each observation by some actor z of a communication between a and b generates zRa, zRb, aRz, or bRz. For the simulation experiments described here, $\theta = 0.25$, $\pi = 0.50$, and $\eta = 0.75$.

In EST research, the relation R is generally taken to be dominance or precedence. Actor a is taken to be in a relation of precedence with respect to actor b if actor a routinely takes or is granted the opportunity to contribute to the task. In this study, the relation is precedence. For additional information regarding EST and the $STPM$ axioms, see [11, 7].

2.2 Individual Decision-making

This study is based on a set of $N = 111$ individual, actual, affirmative decisions, each rendered by exactly one of five types of actor (Teacher, Principal, Nurse, Social Worker, and Counselor) and each corresponding to a single case (indexed by c) corresponding to a distinct, actual, elementary school child. For each such case an affirmative decision was reached regarding the following proposition, denoted $A(c)$: The child described by case c *may* have been physically abused by one of its natural parents. These decision data, which reflect only affirmative decisions, were used to construct a probabilistic belief model for each of the five types of actor.

The purpose of the probabilistic belief model is to represent what belief, based on prior experience as encoded in the decision data, each type of actor would form regarding a previously *unseen* case. Each case c is represented by a vector of seven features: family $INCOME$ and $AFDC$ status (namely, whether the family receives Aid For Dependent Children); the child's AGE, SEX, and $ETHNICITY$; $INHOME$, whether a parent lives in the home of the child, and; $OCCUPATION$, the occupation of the school professional that affirmed $A(c)$.[1]

[1] The case data that formed part of the basis for this study were made available (in part) by the National Data Archive on Child Abuse and Neglect, Cornell University, Ithaca, New York. The data from the Substantiation of Child Abuse and Neglect Re-

Using $N = 111$ feature vectors, probabilistic belief models were constructed for each actor type based on the if-then classification rules produced by the machine learning program RIPPER.[2]

The belief of actor a regarding $A(c)$ is denoted by $B_a(c) \in [0, 1]$. $B_a(c) = 0$ signifies that a denies $A(c)$; $B_a(c) = 1$ signifies that a affirms $A(c)$, and; $B_a(c) = 0.5$ signifies that a is epistemically neutral regarding $A(c)$. Hence, $B_a(c)$ is a continuous variable ranging from utter disbelief through neutrality to complete certainty.

Since the focus of this study is collective rather than individual decision-making regarding $A(c)$, it is necessary to somehow determine the collective decision regarding $A(c)$ from the values of $B_a(c)$ where a ranges over each of the five types of actor. This is accomplished by mapping $B_a(c)$ to $desire_a(c)$, the desire of actor a to make the collective decision regarding $A(c)$ that conforms to his/her belief regarding $A(c)$. The desire of actor a to bring about a collective affirmation or denial of $A(c)$ is represented by $desire_a(c) = B_a(c) - 0.5 \in [-0.5, +0.5]$.[3]

If $desire_a(c) < 0$, this signifies that actor a wishes $A(c)$ to be denied by the group; if $0 < desire_a(c)$, this signifies that a wishes $A(c)$ to be affirmed by the group; if $0 = desire_a(c)$, this signifies that a is neutral regarding the group decision. The larger the absolute value of $desire_a(c)$, the stronger the desire of actor a to bring about a collective decision in accord with $B_a(c)$. In short, desire is assumed to be a function of belief.

2.3 Group Decision-making

The collective decision regarding $A(c)$ is modeled as an influence- weighted sum of $desire_a(c)$ over each type of actor a. Real-time simulation is used to generate the values of of $participation(a)$.[4] The collective decision process is described in greater detail below.

3 The Experimental Conditions

Each set of simulation experiments is identified by the distribution of task-external status within the group. Since each member can be assigned any of

ports Project were originally collected by John Doris and John Eckenrode. Funding support for public distribution was provided by a contract (90-CA-1370) between the National Center on Child Abuse and Neglect and Cornell University. Neither the collector of the original data, funding agency, nor the National Data Archive on Child Abuse and Neglect bears any responsibility for the analyses or interpretations presented here.

[2] RIPPER was chosen in order to obtain easily interpretable rules and because of its relatively high level of performance [**?**].

[3] Note that if $B_a(c) = 0$, then $B_a(c) - 0.5 = -0.05$. Likewise, if $B_a(c) = 1$, then $B_a(c) - 0.5 = +0.05$. Since $B_a(c) \in [0, 1]$, $desire_a(c) \in [-0.5, +0.5]$.

[4] The simulation is written in Java using the channel-based process communication provided by the *JCSP* (*Java Communicating Sequential Processes*) library [9].

three task-external status values, there are $3^5 = 243$ possible experimental conditions based solely on task-external status. In this (preliminary) study, only the four conditions in Table 1 are considered. The entries in Table 1 are defined as

Table 1. Experimental conditions for simulations

Experimental condition	Status condition
0	$SC_0 = HMMLM$
2	$SC_1 = HMMMM$
4	$SC_2 = MMMLM$
6	$SC_3 = HMMHM$

follows. In SC_0, actor 0 has $H(igh)$ task-external status, actors 1, 2 and 4 have $M(edium)$ task-external status, and actor 3 has $L(ow)$ task-external status, and so on. The task-external status of each actor is based on: the credentials held by each, and; the amount of resources available to each actor [12].[5]

4 The Quantities of Interest

Each run of the simulation represents one group meeting. In each such run, $addressed(a, b)$, the *number* of times a addressed b is recorded.[6] The value of $addressed(a, b)$ is determined by a pseudo-random number generation process based on the simulation parameters π, η, and θ. Actors communicate dyadically until 376 dyadic communication events have occurred, thereby simulating a meeting of 40 minutes in duration.[7] In turn, a's level of participation during that one meeting is defined as (the proportion)

$$participation(a) = \sum_{b=1}^{5} addressed(a, b) / \sum_{z=1}^{5} \sum_{b=1}^{5} addressed(z, b) \qquad (1)$$

A number of experimental and observational studies over a half century suggest that the influence of an actor in a small, task-oriented, group is highly correlated with the quantity of their participation.[8] Those who participate most/least

[5] From a *psychological* social psychology viewpoint, it is quite right to consider personality as a potential determiner of *task-internal* status. From the perspective of *sociological* social psychology [6, pp. ix–xiii], however, it is the task-internal behaviors generated by such personality characteristics that are of interest in explaining the emergence of dominance or precedence structures in small, task-oriented, groups.

[6] The abstractness of the $STPM$ and the rudimentary nature of the simulation engine is such that there is no mechanism for modeling the dependence of $addressed(a, b)$ upon c.

[7] The number 376 is a normalized value derived from observational data presented in [8].

[8] For a brief review of this literature, see [11].

in a small, task-oriented, group generally have the most/ least influence. Not all participation, however, is influential.[9]

I adopt a "rich get richer, poor get poorer" view of influence in relation to participation with

$$influence(a) = participation(a)^2 / \sum_{z=1}^{5} participation(z)^2. \quad (2)$$

Note that the values of $influence(a)$ obtained via equation (2) preserve the order relations that exist among the values of $participation(a)$.

For each experimental condition in Table 1, the simulation consists of 100 batches of 20 independent runs. Each batch provides a sample mean for the population parameter $participation(a)$. Regardless of how $participation(a)$ is distributed, the Central Limit Theorem implies that the sample means are themselves normally distributed about $\mathcal{E}[participation(a)]$, the expected (or mean) value of $participation(a)$.

$CD(c)$, the collective decision of a task-oriented group concerning $A(c)$, is modeled as follows.

$$CD(c) = \begin{cases} 0 \text{ if } \sum_{a=1}^{5} influence(a) \cdot desire_a(c) \leq 0 \\ 1 \text{ otherwise} \end{cases}$$

$CD(c) = 0$ signifies that $A(c)$ is denied; $CD(c) = 1$ signifies that $A(c)$ is affirmed. Since $influence(a)$ is a random variable, so too is $CD(c)$. Under each experimental condition, the estimated expected value of $CD(c)$ is

$$\hat{\mathcal{E}}[CD(c)] = \sum_{a=1}^{5} desire_a(c) \cdot \hat{\mathcal{E}}[influence(a)] \quad (3)$$

4.1 Intermediate Values

Table 2 gives the estimated value of, and confidence limits for, the expected value of $participation(a)$ along with $\hat{\mathcal{E}}[influence(a)]$, the estimated expected value of $influence(a)$. The latter is computed from equation (2) by replacing $participation(a)$ with its estimated expected value.[10]

Within each experimental condition, the values of $\hat{\mathcal{E}}[participation(a)]$ are ordinally consistent with the predicted outcomes: participation is positively correlated with status rank. Moreover, as can be determined by comparing confidence intervals, a number of these differences in estimated expected participation are statistically significant at the $\alpha = 0.05$ level. *Across* experimental conditions,

[9] An obvious limitation of the model is that the *quality* of participation is not considered, a task for future work: As noted by a reviewer, assessing the quality of participation is no easy task.

[10] Although the expected value of the LHS of equation (2) is not strictly equal to the expected value of the RHS, for reasons of expediency I estimate $\mathcal{E}[influence(a)]$ as if it were, ignoring bias.

Table 2. Participation and Influence

Experimental condition	Actor a	Lower limit	$\hat{\mathcal{E}}[participation(a)]$	Upper limit	$\hat{\mathcal{E}}[influence(a)]$
0	0	0.2381	0.2399	0.2417	0.2858
	1	0.1996	0.2011	0.2027	0.2009
	2	0.1986	0.2005	0.2024	0.1997
	3	0.1514	0.1527	0.1541	0.1158
	4	0.1976	0.1996	0.2017	0.1979
2	0	0.2387	0.2405	0.2422	0.2896
	1	0.1880	0.1900	0.1921	0.1808
	2	0.1838	0.1859	0.1879	0.1730
	3	0.1864	0.1886	0.1909	0.1781
	4	0.1866	0.1888	0.1910	0.1785
4	0	0.2083	0.2103	0.2124	0.2210
	1	0.2089	0.2110	0.2130	0.2225
	2	0.2077	0.2098	0.2119	0.2199
	3	0.1508	0.1524	0.1539	0.1161
	4	0.2081	0.2101	0.2121	0.2206
6	0	0.2281	0.2298	0.2315	0.2628
	1	0.1755	0.1773	0.1791	0.1564
	2	0.1754	0.1774	0.1795	0.1566
	3	0.2311	0.2330	0.2348	0.2701
	4	0.1740	0.1760	0.1780	0.1541

$\hat{\mathcal{E}}[participation(a)]$ by the highest status individual(s) is not uniform. Experimental condition 2 has the least status differentiation and actor 0 has the highest participation across the four experimental conditions considered. Although the differences between the proportions are small in absolute terms, a number of the differences in the estimated expected value of participation *across* experimental conditions are statistically significant at the $\alpha = 0.05$ level. As predicted, status matters in relation to participation.

5 Findings

The question now is whether differences in influence lead to systematic differences in the collective decision outcomes. The expected number of negative and affirmative collective decisions under each experimental condition, shown in Table 3, were obtained from equation (3) based on values of $desire_a(c)$ (not shown here) and the values of $\hat{\mathcal{E}}[influence(a)]$ shown in Table 2.[11] Table 3, which is computed using equation (3), suggests that actor 0 (of type School Principal) and actor 3 (of type Teacher) hold opposing beliefs regarding some cases, so that a change in their relative task-external status leads, via a change in their

[11] Since the status distributions across experimental conditions differ only with respect to actors 0 and 3, only the status of those actors is displayed in Table 3.

Table 3. Collective Decisions regarding $A(c)$ over all c

Experimental condition	Actor 0 status	Actor 3 status	Total such that $\hat{\mathcal{E}}[CD(c)] = 1$	Total such that $\hat{\mathcal{E}}[CD(c)] = 0$
0	H	L	44	67
2	H	M	49	62
4	M	L	47	64
6	H	H	56	55

participation and influence, to a change in the number of affirmative collective decisions.

The smallest number of affirmative decisions occurs in experimental condition 0. That is increased if: the task-external status of actor 3 is raised to M or H, or; the task-external status of actor 0 is lowered to M. With respect to experimental conditions 2 and 4, the largest effect is obtained by raising the status of actor 3 rather than simply lowering the status of actor 0. As indicated via experimental condition 6, raising the task-external status of actor 3 from L to H counteracts the high task-external status of actor 0.

For each pair of experimental conditions shown in Table 3, the null hypothesis of equal proportions (of affirmative and negative decisions) was evaluated using Fisher's two-sided exact test (which is a more precise cousin of the χ^2 test) [4, pp. 307]. Out of a series of pairwise comparisons, the smallest p-value of 0.138 is obtained when comparing experimental conditions 0 and 6. So, although there is a systematic difference in the number of the number of affirmative collective decisions obtained in conditions 0 and 6, it is not enough to generate a statistically significant difference.[12]

The discussion thus far has focused entirely on how variations in *individual* status affect the collective decision. Another way of considering the situation is to ask how the weight given to case features varies with experimental condition, which in this paper reduces to the status distribution. Since the feature values associated with each case are, with the probable exception of ethnicity, objective characteristics, a shift in the distribution of influence *may* amount to a change in the importance accorded to these features by the group.

From the data in Table 3, $P_0 = 44/111 = 0.396$ and $P_6 = 56/111 = 0.505$. A shift in the status of actor 3 from L to H is associated with a greater number of expected affirmative collective decisions. To help understand this shift, a logistic regression was performed with the $\hat{\mathcal{E}}[CD(c)]$ as the dependent variable and $INCOME$, $AFDC$, AGE, SEX, $ETHNICITY$, $INHOME$ as the explanatory variables. Since the point is to understand the shift as a function of the a group property, the task-external status composition, the feature $OCCUPATION$ is not included. A logistic regression was performed because

[12] It appears, based on Krippendorff's α, that the level of agreement amongst the actors is not high enough to explain the lack of statistical significance of Fisher's test.

the $\hat{\mathcal{E}}[CD(c)]$ is a binary variable, a situation for which a linear regression is generally inappropriate.

In order to give a qualitative portrayal of differences in the relative importance accorded to the explanatory variables under experimental conditions 0 and 6, I construct a visualization of how $P\{\hat{\mathcal{E}}[CD(c)] = 1\}$, the probability that $\hat{\mathcal{E}}[CD(c)] = 1$, changes in response to a unit change in a single (*standardized*) explanatory variable when all other such explanatory variables are regarded as fixed.

As indicated by Figure 1, the value of ΔP (an abbreviation for $P\{\hat{\mathcal{E}}[CD(c)] = 1\}$) depends on the value of P . I am interested in the values of ΔP when $P = P_0$ and $P = P_6$.[13]

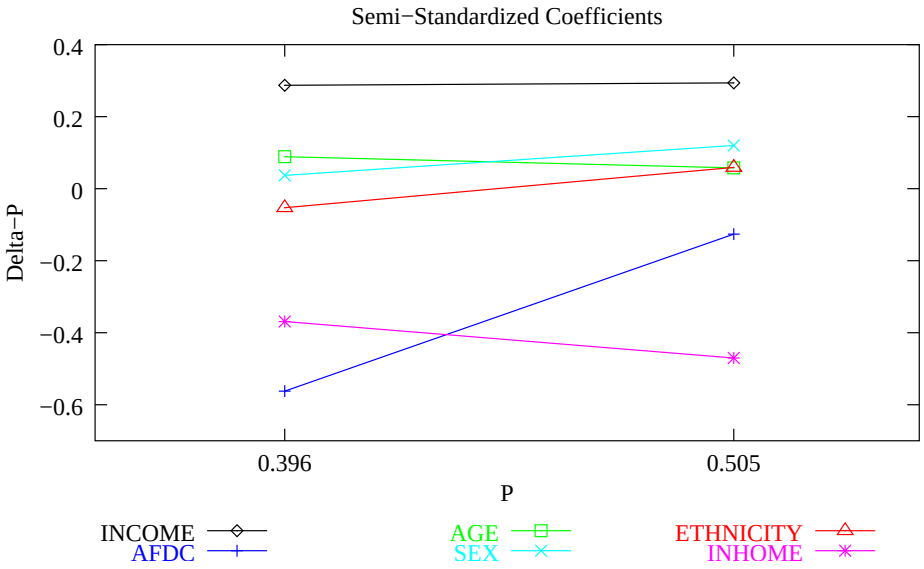

Fig. 1. $\Delta P\{\hat{\mathcal{E}}[CD(c)] = 1\} = f(\sigma, P)$

For experimental condition 0, P is most sensitive to change in $AFDC$. The point associated with $AFDC$ is furthest away from 0 and is negative, indicating that change in $AFDC$ reduces P. A smaller reduction in P results from change in $INHOME$, followed closely by a positive change in P due to $INCOME$. The effects of AGE, SEX and $ETHNICITY$ are relatively small as indicated by the proximity to the line $\Delta P = 0$.

[13] The plot of ΔP is based on *semi-* standardized (logistic) regression coefficients (termed semi-standardized because the outcome variable is *not* standardized). ΔP lies in the interval $[-1, 1]$ and represents the maximum possible change in P that can result from a change of one standard deviation in a particular explanatory variable. For a full discussion of semi-standardized coefficients in logistic regression, see [5].

For experimental condition 6, P is most sensitive to change in $INHOME$. The point associated with $INHOME$ is furthest away from 0 and is negative, indicating that change in $INHOME$ reduces P. $INCOME$ exerts a lesser, positive, influence on P, while the effect of $AFDC$ is negative and approximately half the magnitude of that exerted by $INCOME$. Once again, the effects of AGE, SEX, and $ETHNICITY$ are relatively small.

Whereas P is relatively more sensitive to changes in $AFDC$ and $INHOME$ under both experimental conditions, their ordering is different. Raising the status of actor 3 to that of actor 0 in experimental condition 6 reverses the importance assigned by the group to $AFDC$ and $INHOME$ under experimental condition 0. In effect, the importance assigned by the group to these explanatory variables depends on the task-external status distribution within the group.

6 Related Work

Recent work on social collusion models how relationships are linguistically constituted on the basis of interpersonal characteristics [2]. Although power is identified as an important, longer-term, dimension of social relations, it is not explicitly modeled. The chief aim of the work described here is to predict the influence of actors, an attribute correlated with power.

In a study of cooperation in peer to peer networks, the authors observe that free-riding is less an economic, and more a social-psychological, issue [1]. Their work describes an adaptive agent that models user interests and the social relationships amongst users using reinforcement learning techniques. Like the work on social collusion, the work on peer cooperation focuses on the emergence of interpersonal relations. In contrast, the work described here focuses on how task-external status (often a matter of stereotyping) conditions (but does not determine) the emergence of social order in a small, task-oriented, group. The work described here may be complementary to that on peer-to-peer networks.

In the initial presentation of results obtained from the $STPM$, attention focused primarily on inequality of participation and the correlation of status and participation as a function of θ, π, and η over a wide range of values [7]. Whether differences due to status were statistically significant was not addressed. In [**?**], the $STPM$ was used to explore inequality of participation and the correlation of status and participation, but over a small region of the (θ, π, η) parameter space. Several statistical tests were performed, but influence was not modeled.

7 Summary

This paper describes a simulation approach to the study of task- external status effects in small, task-oriented, groups, It is the first work I know of where influence is modeled as a function of participation. Although status differences were statistically significant in one comparison at only the $\alpha = 0.138$ level, this does suggest that collective decision- making sometimes depends in part on the status of group members *before* they begin deliberating. Although it is important to

examine the behavior of the $STPM$ simulations under a wider set of parameters, it is equally important to augment the bare notion that actor a addresses b with a representation of socio-linguistic and other behavior.

Attention has recently been drawn to possibility of decentralized user models in which user data fragments are dispersed among various devices, services and agents. In distance education, it is important that task-external status differences not shape discussion or deliberation outcomes. It may, then, be useful to regard the $STPM$ (and other models) as a special sort of data, a template, useful for requesting and interpreting data pertaining to interaction in a small, task-oriented, group.

References

1. Bretzke, H., Vassileva, J.: Motivating cooperation on peer-to-peer networks. In: User Modeling 2003, Proceedings of the 9th International Conference, Johnstown, PA (2003) 218-227
2. Cassell, J., Bickmore, T.: Negotiated collusion: Modeling social language and its relationship effects in intelligent agents. *User Modeling and User-Adapted Interaction* **13** (2004) 89-132
3. Cohen, William: Fast Effective Rule Induction. In: Machine Learning, Proceedings of the 12th International Conference, Lake Tahoe, CA (1995)
4. Collett, D.: *Modeling binary data.* Chapman and Hall, Boca Raton, FL (2003)
5. Kaufman, R.: Computing effects in dichotomous logistic regression: A variety of standardized coefficients. *Social Science Quarterly* **77** (1996) 90-109
6. Lawler, E. J., Markovsky, B.: Sociological social psychology: A preface. In Lawler, E. J., Markovsky, B. eds.: *Social psychology of groups: A reader.* JAI Press, Greenwich, CT (1993)
7. Skvoretz, J., Fararo, T.J.: Status and participation in task groups: A dynamic network model. *American Journal of Sociology* **101** (1996) 1366-1414
8. Smith-Lovin, L., Skvoretz, J. V., Hudson, C. G. : Status and participation in six-person groups: A test of Skvoretz's comparative status model. *Social Forces* **64** (1986) 992-1005
9. Welch, P. H. , Austin, P. D.: Java Communicating Sequential Processes (JCSP) Library, Release Candidate 4. University of Kent, Canterbury England (2003)
10. Wilson, R.: Simulating classroom interaction using communicating process architecture. In: Proceedings of the 2001 International Conference on Parallel and Distributed Processing Techniques and Applications, Las Vegas, NV (2001)
11. Wilson, R.: Modeling task-oriented discussion groups. In: Proceedings of the 9th International Conference, UM2003, Johnstown, PA (2003) 248-257
12. Wright, E. O.: The comparative project on class structure and class consciousness. *Acta Sociologica* **32** (1989) 3-22

Location and Activity Modelling in Intelligent Environments

Robert Whitaker, Judy Kay

School of Information Technologies,
University of Sydney
{rwhitake,judy}@it.usyd.edu.au

Abstract. This paper describes ULAP, a framework for scrutable modeling and prediction of people's locations and activities, based upon a diverse collection of sensors, with varying reliability. It supports transformation and aggregation of sensor data, using this to build individual user models of location and activity. We propose an approach to indicate the certainty of predictions about users based upon unobtrusive data for location: it can be provided to applications and also serves as a form of explanation to users. We use this to report experiments involving 32 users, each with varying amounts of historic sensor data for machine activity, formal schedule and Bluetooth device detections. This is combined with group membership.

1 Introduction

Intelligent environments with ubiquitous computing need to exploit the large amounts of data from many, diverse sensors to build user models so that these can serve personalized applications. There are several approaches to modeling user location, for example Active Badge [1], BlueStar [2] and Lancaster Guide [3]. There has also been some recent work in machine learning to predict a user's future location, such as the Assisted Cognition project [4]. Corresponding work on modeling user's activity has had less attention, although there was early work by Orwant [5] and more recent work by Koile et al. [6]. We would like to go beyond these, combining sensor information about location and activity to model and predict both at the time of a request and into the future.

We explain our motivation in terms of the *Boris's Smart Office Door* Scenario; it was introduced in [7]. Boris is an academic, who always carries a Bluetooth enabled PDA. Natasha, a student, comes to his office to meet him. Unfortunately, he is not there. However, his smart door provides an interface which enables Natasha to request help in meeting him. The interface responds, according to Boris's context. Example responses include: Boris is nearby and interruptible so *Boris's Smart Office Door* sends him a message and he comes back to his office to talk with her; Boris is at a seminar and not interruptible but normally returns to his office after seminars so *Boris's Smart Office Door* tells Natasha he is likely to be here in 20 minutes (after the seminar); Boris is at home so *Boris's Smart Office Door* tells Natasha he is unavailable today.

We have determined the following requirements for a framework to support applications like *Boris's Smart Office Door*. It should: support modeling and prediction of location and activity over time, with flexibility in the time granularity of modeling; support multiple applications; make use of multiple, heterogeneous sensors; be easy to manage new, lost or altered sensors; support scrutability, meaning that it can explain its reasoning; protect the user's privacy through a permission system; make use of data for individuals and groups.

Section 2 gives an overview of ULAP and Section 3 describes our approach to representing certainty. We use this in the Section 4 report of evaluation. Section 5 has related work Section 6 has conclusions and future work.

2 ULAP Framework

The ULAP (User Location and Prediction) framework is shown in Figure 1. Its design has been influenced by the architecture of systems like Doppelganger [5], Web Guide [8], and MyPlace [7]. ULAP has three core components: the environment; the core of ULAP; and the applications which use it.

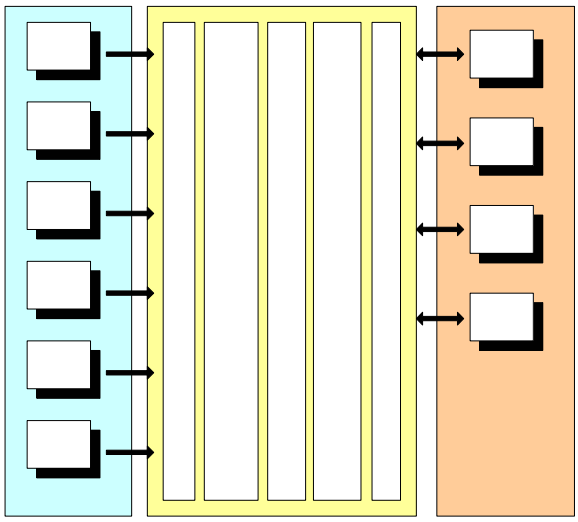

Fig. 1. ULAP Framework

The environment, shown at the left of Figure 1, can include arbitrary numbers of heterogeneous devices/sensors. In our implementation, there were six different types of sensors. These sensors and their purpose are summarized in Table 1. To ensure decoupling of the sensors from the ULAP core, we use publish/subscribe messaging to transmit data from the sensor to the central framework, as was done in MyPlace [7], although that work integrated just two sensors types. The ULAP approach enables sensors to collect data which is forwarded to all applications with subscriptions at the

server. The sensor software does not need to know about those applications, decoupling the sensors, and hence, the environment, from the core framework.

Table 1. Summary of the different types of sensors

Sensor Type	Description/Purpose
BSpy	A Bluetooth based indoor positioning system. Determines location by querying all the Bluetooth enabled devices in range of its sensors.
BlueStar	Uses a combination of indoor and outdoor positioning systems to determine a person's location. The indoor positioning systems used Bluetooth technology.
Windows Activity	Focused on collecting information on the processes and machine a user was using at regular time intervals. Determined if a user was active at the machine or not through analyzing the times between keyboard and mouse events.
Login Sensor	Aimed at tracking a user's machine sessions on a network. It records a user's session information as well as the machine they are logged onto. This information can then be used to determine the location of the user.
Finger Sensor	This sensor collected location and activity information through the use of the *who* and *finger* commands. Location was determined based on the machine name, and activity by the value of the idle field from the finger command.
PDA	Enables a user to log activities and whether interruptible or not.

We now describe the elements of the ULAP core. Leftmost in Figure 1 is the data converter/filter. This must deal with two tasks: aggregation of data from multiple sensors and the conversion of data to a form suitable for the user models.

First consider aggregation. Each sensor can record different types of data and can represent the same data in different ways. For example, the BSpy sensor represents a location using the MAC address of the sensor (00:01:0E0:41:E0:10), while the login sensor represents the location as the machine name (pg-g62-1). In such cases, data from the two sensors cannot be merged directly to give the correct symbolic location[1]. ULAP must map from the raw values from each sensor to consistent symbolic values.

The importance of this issue may not immediately be obvious: much of this functionality could be handled inside the user modeling component or by an application using the user model. However, this is impossible where sensors have different ways to identify users. The data converter/filter component must ensure data is added to the correct user model. It maps the user ID for each sensor to the internal representation used by the ULAP framework.

A similar problem relates to handling multiple devices for the same user. For example, the BSpy sensor identifies users by the MAC address of their device. Where a user carries two devices, a phone and a PDA, both must map to the same symbolic value.

The implementation of this process is based on an approach similar to that of XSLT transformations of XML documents. It builds an internal representation of the XML formatted conversion file. Using this representation it attempts to find an appropriate mapping and apply the conversion. If no mapping is found the original raw value is used.

As data is collected, it must be stored and modeled. This component of the ULAP core uses PersonisLite, a light weight version of Personis [9]. The user model has two contexts, one for the modeled components of location and the other for components of the user's activities. This part of the framework supports group modelling, by

[1] Symbolic location refers to the human representation of a location eg. the name of a room

dynamically generating required group models at runtime, based upon the individual models for each member of the group.

The next part of the ULAP core is the resolvers: these are responsible for interpreting sensor evidence within the user models. Resolvers are selected, at runtime by the application. Different resolvers provide variable granularity of location and activity prediction, as needed for the different subcases of the scenario.

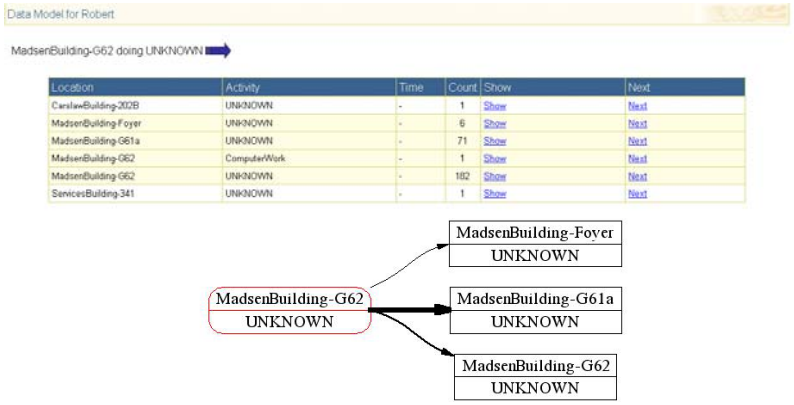

Fig. 2. Example of the ULAP generic interface supporting user scrutiny of results.

ULAP predictions of the user's future location and activity are based upon Markov Chain models, a choice based on its simplicity and the potential for intuitive explanations of the system operation. This means that ULAP can enable users to scrutinize the user modeling processes. Each location/activity pair is represented as a node and possible path in the chain.

The rightmost part of the ULAP core shown in Figure 1 is the interface support enabling the user to see the user model. The Markov model gives a natural visualization of the system's reasoning on the person's movement between locations and activities. An example of a model visualization is shown at the bottom of Figure 2. ULAP supports variable length models. The figure also shows the interface that enables a user to dynamically iterate through the models to see how predictions were determined and to explore additional predictions into the future.

The last main part of the ULAP architecture is the applications, such as *Boris's Smart Office Door*. Shown at the right of Figure 1, three applications we have built to evaluate ULAP are: first, ULAP Modeler, for individual users; second, Group Modeler; and third, Last Location/Last Activity, which query the user's current or last known location and activity (as a basis for prediction into the future as in the scenario where Boris was at a seminar).

The framework has been implementated in a combination of Perl and Python scripts which interact with and manipulate the data stored in the user models. Through the use of system hooks it was possible to monitor mouse and keyboard events to ensure accuracy in the assumptions made by the activity sensors for the activity sensors. The scrutable interface is a Perl based web interface which uses *dot* [10].

3 Modeling Certainty and accuracy

Ideally, we would have had a set of gold standard training and test data: then we could have used various resolvers to query the user models and then compare the results with the known correct result. Indeed, we built tools to collect such data, based upon users maintaining a log of their actual location/activity. Various paper schemes as well as a PDA application were tried. It is unsurprising that people found it too difficult to remember to keep the record (or too irritating to be reminded).

Accordingly, we decided that a different approach was needed. Our approach was partly motivated by our goal of scrutability: we wanted to be able to inform both users and applications of the certainty of a prediction. We identified two elements of this:

- The *consistency* of the available evidence;
- The *nature* of the evidence available.

To determine a consistency value for a prediction, ULAP calculates *{wi}*, a set of weights, where each *wi* is the weight of the evidence for the *i*-th location/activity supported by any of the evidence. ULAP then determines *max{wi}*, meaning that *i* is the value with the highest weight. This value is the result of the query. Its consistency is calculated as *max{wi}/sum{wi}*. If there is no evidence for a query, we return a consistency value is 0. With one piece of evidence, it is 1.0.

This can be calculated at the time of the user model query. Then, ULAP applies the appropriate location/activity granularity. So, for example, if an application asks if the user is interruptible or not, there are two values and each piece of evidence is interpreted to contribute to the weight of support for one. If, on the other hand, a query specifies a resolver with several location/activity values, ULAP calculates the total evidence weight for each of these. There are many ways to calculate the weights. A review of a range of such algorithms has been described for ubiquitous computing [11]; any of these could be applied within ULAP. Notably, since we want to deal with multiple sensors of varying reliability, an algorithm can exploit knowledge to adjust the weight according to sensor reliability.

To illustrate the process, suppose 180 pieces of evidence support location A and 20 support location B. An algorithm that treats all evidence equally returns the value A, with consistency 90%. Taking another example, if there are 10 equal-weight pieces of evidence for each of 20 different location/activity values, each is equally likely. The resolver returns one of them, with accuracy 5%.

Clearly, there are serious limitations to this consistency measure. The second element of certainty relates to the *nature of the evidence* and has to help deal with this. For example, consider the case in the paragraph above for locations A and B. One very simple indication is the total number of pieces of evidence. This measure is what we have used.

In summary, in lieu of an accuracy measure we use consistency and the amount of evidence. This is clearly inferior to a measure of true accuracy, calculated by comparing a ULAP prediction against a known correct result. However, in our experiments, that was unavailable. Moreover, in general, it will be important for user modeling predictions for ubiquitous applications to include a prediction of the accuracy of the result [12]. So, it is important to define a practical way to indicate the certainty of a prediction, as our approach does.

4 Evaluation

Our evaluation tested the effectiveness of the ULAP framework by implementing it and then using it to build a range of models. We now report its use in:

- modeling individual users, based upon a variety of sensors for location and activity, with historic data used to support predictions and comparing the effect on certainty from the evidence of additional sensors;
- modeling groups by aggregating individual models, comparing the effect on certainty of predictions, where this had the potential to provide predictions for individuals even when no sensor data was available for them but there was data for people in the same group.

As already discussed, individual model certainty is based upon consistency and the amount of evidence for predictions. This section summarizes results for multiple heterogeneous sensors, individual and group modeling. For fuller results as well as scalability experiments, see [13].

Our experiments have been based upon data for 32 users. A summary of the data for four of the more interesting users is summarized in Table 2. Data was collected over 4 months for the BlueStar (Bluetooth) sensor types, and 6 weeks for the other sensor types. This is of a similar order to much of the published work, such as the Assisted Cognition project [4, 14-17] which had 6 months of a single data type, GPS, to model an individual's movements around a large city. We used this to build individual models.

Table 2. Details sensor readings or detections for 7 users with relatively rich collections.

User	Number of detections recorded						
	BSpy	BlueStar	Login	Activity	Finger	PDA	Timetable
A1	4,819	-	285	5,717	5,111	250	YES
B1	6,464	-	0	-	10,939	-	-
E1	-	-	159	5987	131	-	YES
H1	-	163,392	0	-	0	-	-

Figure 3 shows the contrasting levels of consistency in two extreme cases. The graph on the left is for User H1 and is built from 163,392 pieces of BlueStar data collected over four months, covering every hour of each day of the week. Consistency values less than 1.0 are due to detection of H1 by multiple sensors at different locations. This graph on the right is for User A1, based upon 16,182 pieces of sensor evidence, representing data collected over each hour of the week. The zero points occurred when there was no data for the user. A comparison of these graphs shows that both return similar consistency readings, even though in the right hand graph we have increased the number and type of sensors used, as well as increasing the number of possible of combinations a user can be detected in a single hour through the observation of activity in addition to location.

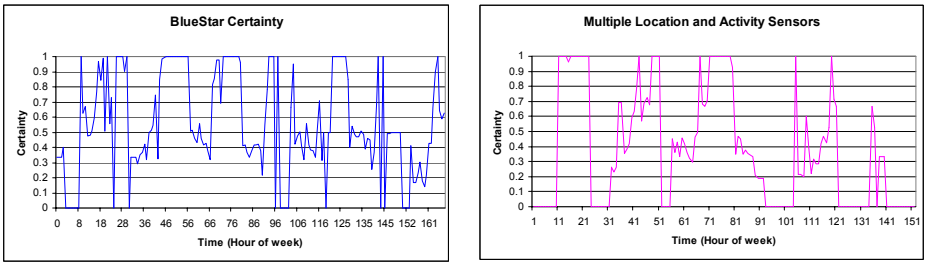

Fig. 3. The graph on the left shows the consistency of models based on 4 months of BlueStar data. Compare this with the graph on the right, which shows consistency of 6 weeks of data collected from multiple heterogeneous sensors.

Figure 4 indicates the relative effect of *activity* sensors in addition to multiple location sensors. The left graph, for user A1's location alone tends to have consistency around 50% for each of the 5 days of the week and no other data. The right graph is for the same user with activity sensors as well. This visually gives a higher consistency. There are many reasons for these differences: the types of sensors, activity sensors usually have a finer location granularity; and the use of additional data captured by these sensors when determining certainty.

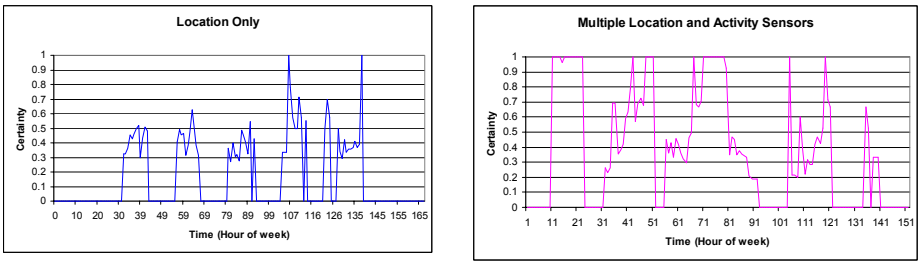

Fig. 4. The graph on the left shows consistency with multiple location sensors. That on the right also has activity sensor data.

In this evaluation, models were constructed for a range of groups of people. Using this calculation time periods where the user mainly performs one event will clearly stand out through a certainty close to one. This can then be compared to those times when many different events have been observed over the user, in this case the certainty will be lower dependent on the number of different events seen and how often each event was observed. We now look at two of those profiles in detail with those being: the profile of a university academic; and that of honors students teaching various courses.

Figure 5 shows the consistency graph for User B1, a university academic. As shown in Table 2, their model is based on substantial data sets from two sources, BSpy and Finger. This person also tends to keep a fairly consistent schedule over the four months: for example the very consistent period around hour 70 of the week is their research group weekly seminar and other meetings. When shown this graph, B1 could identify their various regular activities in the week.

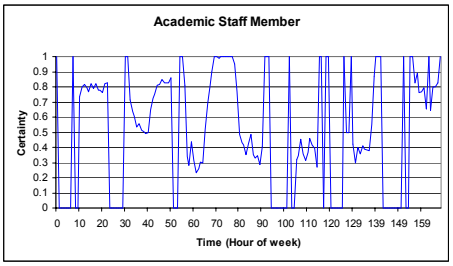

Fig. 5. Prediction consistency for B1

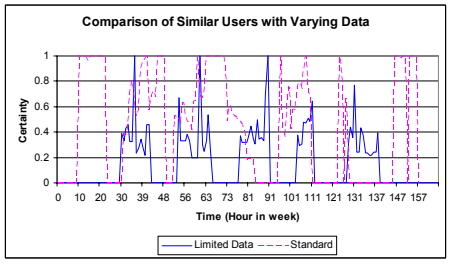

Fig. 6. Consistency for A1 compared with E1.

Figure 6 shows consistency measures for two Honours students, A1 in broken lines and E1 in solid lines. Both have similar schedules but, as can be seen from Table 2, A1 had six sources of sensor evidence where E1 had just four. Notably, A1 had Bspy data but E1 did not. E1, with limited data has consistency values around 0.4 and the five days of the week can be seen clearly. These trends are quite strong, taking account of the 6 week period that provides them. For A1, there are many more periods where predictions have higher consistency, including periods on weekends and nights.

The similarity of the two users of Figure 6 suggests the potential value of exploiting group membership or user similarity to support predictions even for users for whom we have no data. We performed group modeling experiments; these are similar to communities described in Doppelganger [5] although this work does not report results of user experiments as we do below. The group modeling functionality allows a person to be associated with every relevant group. So, for example, an Honours student who tutors and has a desk in Lab 1 can be assigned to multiple groups: Honours which includes people in many labs, tutors which overlaps the Honours group and includes others, Lab 1 group which includes students and research staff in that lab. Table 3 shows the groups identified for experiments.

Table 3. Number of detections per group from each sensor.

Type	Group	Number of detections recorded						
		BSpy	BlueStar	Login	Activity	Finger	PDA	Timetable
Hons	Honours	4819	0	20223	17873	11428	250	2
	Hons Group 1	4819	0	1633	16341	5788	250	2
Tutors	Tutors	10903	0	10009	16341	10564	250	2
	Tutors SOFT2001	4819	0	586	11704	5413	250	2

As seen in the left hand graph of Figure 7 a substantial confidence improvement was obtained for most time periods, as the number of conflicts or possible locations for each time period had been reduced. However, through the modeling of tutors for one particular course no substantial certainty improvement could be gained, nor any conclusive prediction be made about this group because of the group diversity. The certainty results can be seen in the right hand graph of Figure 7.

To identify useful groupings, we created group models which systematically explored each grouping. We then used the consistency measure as a basis for selecting useful groupings. This identified groupings that were unhelpful, such as that

of tutors, where different people are allocated to different classes, meaning that data for one person is generally not consistent with data for others in the group.

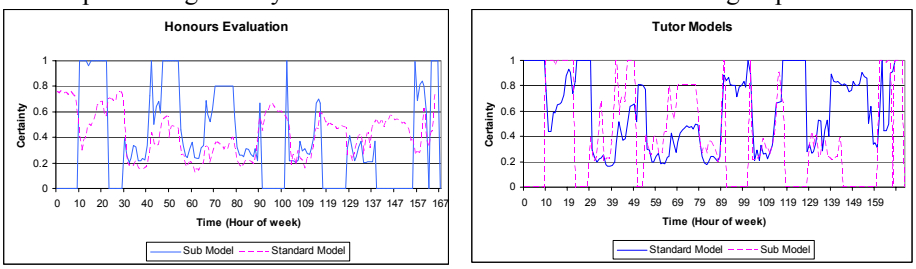

Fig. 7. In the Left graph shows use of subgrouping consistency. In the right graph, this approach was not successful because of subgroup diversity.

4 Related Work

Two projects were particularly important for the design of ULAP. Doppelganger [5] also aimed for a general framework for gathering and processing heterogeneous sensor data and community modeling, but had a different architecture and did not report results of experiments for long term user data. The more recent Assisted Cognition Project [4] models movement paths to assist the mentally disabled. One of its prototype systems, the Activity Compass [14], uses PDA and GPS location sensors. A second project is an application, called Opportunity Knocks [16], designed to run on a mobile phone, models a person's path in a city based on GPS data. There has been some work in using activity sensors, such as Activity Zones [6] and considerable work on location sensing, such as Active Badge [1], Lancaster Guide [3], Web Guide Project [8] and Multiple User Detection [18]. ULAP has explored a different dimension of the problem of modeling user location and activity, with a focus on far more heterogeneity of sensors than is the case in these projects. Several others have also explored the use of Markov models, for example, Assisted Cognition [4], Multiple User Detection [18] and Doppelganger. And there has been work on other learning approaches, for example Web Guide Project [8], Assisted Cognition [4] as well as Doppelganger. Importantly, at this stage in the area of location and activity modeling much of the evaluation has been based upon synthetic data or special test data. Other work that has collected authentic sensor data for normal or near normal users has been done in projects like MyPlace [7], Doppelganger [5], Activity Zones [6], Assisted Cognition [4] and Multiple User Detection [18]. The scale, diversity and time period of our sensor data is broader than these projects.

5 Conclusion

This report has described a framework for modeling location and activity based ondata collected from ubiquitous environments. We demonstrated the effectiveness of

this framework through its implementation and analysis of the models generated by it. We have reported consistency results demonstrating ULAP's ability to refine its model by using multiple heterogeneous sensors and the modeling of groups.

This work provided an initial investigation into the modeling and prediction of location and activity information for an individual and group. The implementation and evaluation of a framework is the first step to the development and support of personalized applications for the user and their environments.

6 References

1. Want, R., et al., *The Active Badge Location System.* ACM Transactions on Information Systems, 1992(January 1992).
2. Quigley, A., et al. *BlueStar, a privacy centric location aware system*. in *IEEE Position, Location and Navigation Symposium 2004 (PLANS 2004)*. 2004. Monterey CA, USA.
3. Distributed Multimedia Research Group, *The Guide Project*. 2004.
4. Kautz, H., et al. *An Overview of the Assisted Cognition Project*. in *AAAI-2002 Workshop on Automation as Caregiver: The Role of Intelligent Technology in Elder Care*. 2002.
5. Orwant, J. *Heterogeneous Learning in the Doppelganger User Modeling System*. in *User Modeling and User-Adapted Interaction*. 1995.
6. Koile, K., et al. *Activity Zones for Context-Aware Computing*. in *Ubicomp*. 2003.
7. Kay, J., B. Kummerfeld, and D. Carmichael, *Consistent Modelling of users, devices and environments in a ubiquitous computing environment*. 2004, University of Sydney.
8. Fink, J. and A. Kobsa, *User Modeling for Personalized City Tours.* Artifical Intelligence Review, 2002. **18**(2002): p. 33-74.
9. Kay, J., B. Kummerfeld, and P. Lauder. *Personis: A Server for User Models*. in *Adaptive Hypertext 2002*. 2002: Springer.
10. Ganser, E., E. Koutsofios, and S. North, *Drawing Graphs with dot*. 2002.
11. Hightower, J. *From Position to Place*. in *2003 Workshop on Location-Aware Computing*. 2003. Seattle, Washington.
12. Antifakos, S., A. Schwaninger, and B. Schiele. *Evaluating the Effects of Displaying Uncertainty in Context-Aware Applications*. in *Ubicomp 2004*. 2004. Nottingham, England.
13. Whitaker, R., *Location and Activity Modelling in Intelligent Environments*, in *School of Information Technologies*. 2004, University of Sydney: Sydney, NSW. p. 131.
14. Patterson, D.J., O. Etzioni, and H. Kautz. *The Activity Compass*. in *First International Workshop on Ubiquitous Computing for Cognitive Aids*. 2002. Gothenberg, Sweden.
15. Patterson, D.J., et al. *Inferring High-Level Behavior from Low-Level Sensors*. in *Fifth International Conference on Ubiquitous Computing*. 2003.
16. Patterson, D.J., et al. *Opportunity Knocks: a System to Provide Cognitive Assistance with Transportation Services*. in *Ubicomp*. 2004. Nottingham, England.
17. Kautz, H., et al., *Foundations of Assisted Cognition Systems*. 2003, UW CSE Technical Report.
18. Ashbrook, D. and T. Starner, *Using GPS to learn significant locations and predict movement across multiple users.* Personal and Ubiquitous Computing, 2003. **7**: p. 275-286.

Decentralized User Modeling with UserML and GUMO

Dominik Heckmann[1], Tim Schwartz[2], Boris Brandherm[2], Alexander Kröner[1]

[1] German Research Center for Artificial Intelligence, Saarbrücken, Germany
[2] Saarland University, Saarbrücken, Germany
{heckmann,kroener}@dfki.de, {schwartz,brandherm}@cs.uni.sb.de

Abstract. We present a new architecture for decentralized user modeling and briefly discuss the user model markup language USERML, the general user model ontology GUMO for the uniform interpretation of decentralized user models, and the integration of ubiquitous applications with the `u2m.org` user model service. The motivation is that ubiquitous evaluation of user behavior with a variety of systems in the web or the physical world might lead to attractive new services.

1 Approach and Architecture

We developed the RDF-based user model exchange language UserML to enable decentralized systems to communicate over user models. The idea is to spread the information among all adaptive systems, either with a mobile device or via ubiquitous networks. `UserML` statements can be arranged and stored in distributed repositories in XML, RDF or SQL. Each mobile and stationary device has an own repository of situational statements, either local or global, dependent on the network accessability. A mobile device can perfectly be integrated via wireless lan or bluetooth into the intelligent environment, while a stationary device could be isolated without network access. The different applications or agents produce or use `UserML` statements

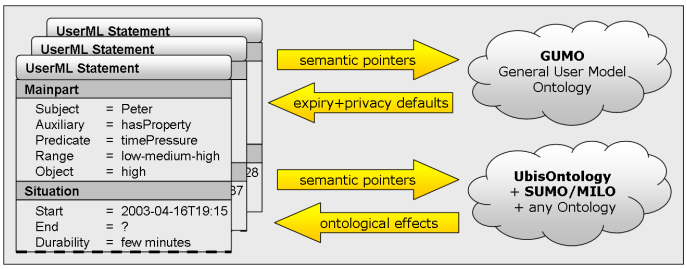

Fig. 1. The syntax-semantics interplay between USERML and GUMO

to represent the user model information. `UserML` forms the syntactic description in the knowledge exchange process, see figure 1. Each concept like the user model auxiliary

hasProperty and the user model dimension timePressure points to a semantical definition of this concept which is either defined in the general user model ontology GUMO, the UbisWorld ontology, which is specialized for ubiquitous computing, or the general SUMO/MILO ontology, see [1]. The merging of partial, decentralized user models is realized by combining the different user model repositories, while the inferential integration is done by filters and conflict resolution strategies as shown in figure 2(b). Figure 2(a) and figure 2(c) show the upward and downward inference from repositories or journals to the user model and vice versa.

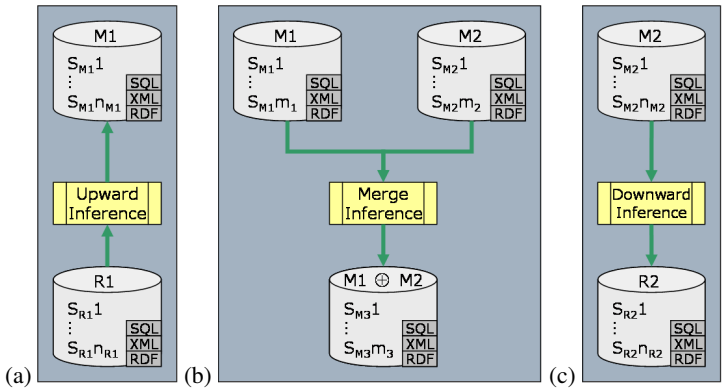

Fig. 2. User model integration with upward inference, merge inference and downward inference

2 The Background of UserML and GUMO

UserML has been introduced in [2] as user model exchange language. A central conceptual idea in USERML's approach is the division of user model dimensions into the three parts auxiliary, predicate and range as shown right below.

$$\texttt{subject}\ \{\ \textit{UserModelDimension}\ \}\ \texttt{object}$$
$$\Downarrow$$
$$\texttt{subject}\ \{\ \texttt{auxiliary},\ \texttt{predicate},\ \texttt{range}\ \}\ \texttt{object}$$

For example, if one wants to say *something about the user's interest in football*, one could divide this so-called *user model dimension* into the auxiliary part *has interest*, the predicate part *football* and the range part *low-medium-high*. Apart from these so called mainpart attributes, further important meta attributes have been identified for the user modeling domain. These are situation (like start, end, durability, location and position), privacy (like key, owner, access, purpose, retention) and explanation (like creator, method, evidence, confidence). UserML statements need not to use all 25 attributes that have been arranged into groups. However each of these have a predefined meaning on which specialized meta-data inference modules work.

The advantage of using UserML to model the user model statements is the uniform syntactical relational data structure that allows apart from the representation in an ontology also the storage of mass data in a database.

GUMO has been introduced in [3]. It is designed according to the approach of dividing basic user model dimensions into triples. The advantage of using GUMO in decentralized settings is the semantical uniformity. Loads of auxiliaries, predicates and ranges have so far been identified and inserted into the ontology that can be inspected with a foldable tree browser at the web page http://www.gumo.org. However, it turned out that actually everything can be a predicate for the auxiliary *hasInterest* or *hasKnowledge*, what leads to a problem if one does not work modularized. The suggested solution is to identify basic user model dimensions on the one hand while leaving the more general world knowledge open for already existing other ontologies on the other hand. Candidates are the general suggested upper merged ontology SUMO, see [1], and the UBISWORLD ontology to model intelligent environments, see http://www.ubisworld.org. This insight leads to a modular approach which forms a key feature of GUMO. A commonly accepted top level ontology for user models could be of great importance for the user modeling research community. But which groups of user dimensions can be identified? In [4] and [5] rough classifications for such categories can be found. Furthermore, this ontology should be represented in a modern semantic web language like OWL and thus via internet be available for all user-adaptive systems at the same time. The major advantage would be the simplification for exchanging interpretable user model information between different user-adaptive systems. differences between existing user modeling systems could be overcome. We are collecting the user's dimensions that are modeled within user- adaptive systems like the *user's heart beat*, the *user's age*, the *user's current position*, the *user's birthplace* or the *user's ability to swim*. Furthermore, the modeling of the user's interests and preferences like *reading poems*, *playing adventure games* or *drinking certain French Bordeaux wines* is analyzed. Identified user model auxiliaries apart from *hasKnowledge* and *hasInterest* are for example *hasBelieve*, *hasPlan*, *hasProperty*, *hasGoal*, *hasPlan* and *hasRegularity*. User model predicates that fit to the auxiliary *"hasProperty"* are called *BasicUserDimensions*. Examples are Emotional States, Characteristics and Personality. The following listing presents the concept *PhysiologicalState* defined as owl:Class. It is defined as a subclass of *BasicUserDimensions*. A class defines a group of individuals that belong together because they share some properties. Classes can be organized in a specialization hierarchy using rdfs:subClassOf.

```
<owl:Class rdf:ID="PhysiologicalState.700016">
  <rdfs:label> Physiological State </rdfs:label>
  <rdfs:subClassOf rdf:resource="#BasicUserDimensions.700002" />
  <gumo:identifier> 700016 </gumo:identifier>
  <gumo:lexicon>state of body or bodily functions</gumo:lexicon>
  <gumo:privacy> high.640033 </gumo:privacy>
  <gumo:website rdf:resource="&GUMO;concept=700016" />
 </owl:Class>
```

Every concept has a unique rdf:ID, that can be resolved into a complete URI. The attribute gumo:privacy defines the default privacy status for this class of user dimensions. The attribute gumo:website points towards a web site, that has its purpose in presenting this ontology concept, to a human reader. The abbreviation &GUMO; is a shortcut for the complete URL to the GUMO ontology in the semantic web. rdf:Desc-

ription. The attribute gumo:expiry provides a default value for the average expiry which carries the qualitative time span of how long the statement is expected to be valid. In most cases when user model dimensions are measured, one has a rough idea about the expected expiry. For instance, emotional states hold normally no longer than 15 minutes, however personality traits won't change within months. Since this qualitative time span is dependent from every user model dimension, it should be defined within GUMO. Some examples of rough expiry-classifications are:

- physiologicalState.heartbeat - can change within seconds
- characteristics.inventive - can change within months
- personality.introvert - can change within years
- demographics.birthplace - can't normally change at all

The idea behind gumo:expiry is that if no new actual value is available on the user model server after a while, one can still work with old values, probably combined with reduced confidence values. The semantic web ontology language OWL allows to construct complex, graph-like hierarchies of user model concepts with multiple-inheritance, which is especially important for ontology integration. For example, *happiness* is defined as rdf:type of the class *EmotionalState* and *FiveBasicEmotions*. The GUMO vocabulary includes gumo:identifier, gumo:expiry, gumo:image, gumo:privacy, gumo:website, gumo:image and gumo:lexicon. To support the distributed construction and refinement of GUMO, we developed a specialized online editor to introduce new concepts, to add their definitions and to transform the information automatically into the required semantic web language. Related ontologies and knowledge-sharing projects have been analyzed. However, a profound investigation of the possible application of student model standardizations into the domain of ubiquitous computing has to be undertaken.

3 Decentralized and Mobile Services and Applications

A user model *service* manages information about users and contributes additional benefit compared to a user model *server*. The u2m.org user model service consists of a set of application-independent servers with a distributed approach for accessing and storing user information, the possibility to exchange and understand data between different applications, as well as adding privacy and transparency to the statements about the user. Applications can retrieve or add information by HTTP requests like:

```
http://www.u2m.org/UbisWorld/UserModelService.php?
subject=Peter&auxiliary=hasProperty&predicate=Happiness
```

We have tested the approach in a MOBILEMUSEUMSGUIDE, see [6], in a POSITIONINGSERVICE, see [7] and in an ALARMMANAGER application, see [8]. The latter one is a notification service for instrumented environments that adapts the presentation of announcements to the user's state of arousal and the user's location. Both are retrieved from the UserML and GUMO enabled user model service. The location is derived from the POSITIONINGSERVICE application. This service runs on the user's PDA and uses infrared beacons and active RFID tags that are installed in the environment to estimate the location of the user which is then send via WiFi to the user model service. Figure 3 shows three identified conceptual situations to decentralized user modeling.

==========

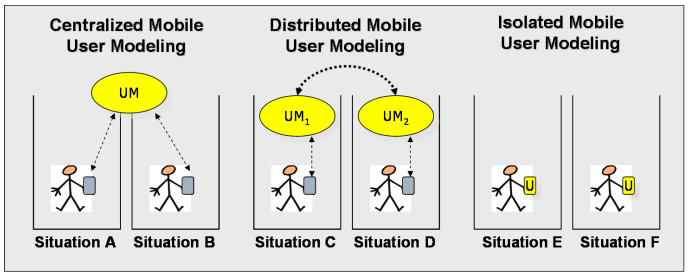

Fig. 3. Concept of Centralized, Distributed and Isolated Mobile User Modeling

Summary & Acknowledgements

We presented architectural modules for decentralized user modeling, especially the user model exchange language `UserML` and the general user model ontology GUMO. We discussed the possibilities for semantic integration and briefly demonstrated the approach with ubiquitous user-adaptive applications. The combination of the GUMO ontology with the exchange language `UserML` together with the decentralized `u2m.org` user model service seems to be promising. This research is supported by the German Ministry of Education and Research (BMB+F) under grant 524-40001-01 IW C03, the project SPECTER and by the German Science Foundation in its Collaborative Research Center on Resource-Adaptive Cognitive Processes, SFB 378, Project EM 4, BAIR.

References

1. Pease, A., Niles, I., Li, J.: The suggested upper merged ontology: A large ontology for the semanticweb and its applications. In: AAAI-2002Workshop on Ontologies and the Semantic Web. Working Notes (2002)
2. Heckmann, D.: Introducing situational statements as an integrating data structure for user modeling, context-awareness and resource-adaptive computing. In: ABIS2003, Karlsruhe, Germany (2003) 283–286
3. Heckmann, D., Brandherm, B., Schmitz, M., Schwartz, T., von Wilamowitz-Moellendorf, B.M.: Gumo - the general user model ontology. In: User Modeling, Edinburgh, Scotland (2005)
4. Jameson, A.: Systems That Adapt to Their Users: An Integrative Perspective. Habil, Saarbrücken, Germany (2001)
5. Kobsa, A.: Generic user modeling systems. User Modelling and User-Adapted Interaction Journal **11** (2001) 49–63
6. Kruppa, M., Heckmann, D., Krüger, A.: Adaptive multimodal presentation of multimedia content in museum scenarios. KI Journal **1** (2005) 56–59
7. Brandherm, B., Schwartz, T.: Geo referenced dynamic bayesian networks for user positioning on mobile systems. In: Proceedings of the International Workshop on Location- and Context-Awareness (LoCA), Munich, Germany (2005)
8. Brandherm, B., Schmitz, M.: Presentation of a modular framework for interpretation of sensor data with dynamic Bayesian networks on mobile devices. In: LWA 2004, Lernen Wissensentdeckung Adaptivität, Berlin, Germany (2004) 9–10

Purpose-based User Modelling in Decentralized Agent and Web-Service Based Environments

Xiaolin Niu, Julita Vassileva, Gordon McCalla [1]

Department of Computer Science
University of Saskatchewan
Saskatoon, Saskatchewan, S7N 5A9 Canada
{xin978, mccalla, jiv}@mail.usask.ca

Abstract. This paper outlines a new approach for decentralized agent based user modelling using a taxonomy of *purposes* that define a variety of context-dependent user modelling processes rather than creating and maintaining a single centralized user modelling server. This approach can be useful in distributed environments where autonomous agents develop user models independently and do not necessarily adhere to a common representation scheme.

1 Decentralized active user modelling

Traditionally user modelling has focused on creating and maintaining a single global description of the user used internally in an application for some purpose defined at design time [1]. Knowledge representation is a key issue in this kind of traditional user modelling. With the emergence of networked applications, user modelling servers have been proposed [2] to store data that can be used to support adaptation in several networked applications. User modelling servers provide a centralized solution: user models are stored in centralized or virtually centralized repository. Even if the user data comes from and serves various applications, the representation of the user model follows a particular centralized schema, which is known in advance to the applications.

However, software systems currently are shifting to web-services, which not only distributed, but also autonomous and often agent-based. The autonomous agents or web services keep user model fragments, which can be used by others only if the services/agents are willing to share the information [4, 8]. These fragments cannot be expected to use the same representation scheme (the same problem arises in distributed databases, see [3]). Even if they wish to do so, the user model fragments come from a range of sources (e.g. raw data, other agents) and are dependent on the context in which they were created, so it would be very hard to ensure consistency in a centralized user model based on these fragments as input. Therefore the focus of user modelling shifts from the collection at one place of as many data about a user as

[1] This work is supported by the Natural Sciences and Engineering Research Council of Canada.

possible to collecting on demand whatever user information is available at the moment from various agents and interpreting it for a particular purpose. This is called *active* user modelling [4].

2. Purpose-based user modelling

This paper presents briefly a purpose-based approach to active user modeling [5], which is aimed at defining a taxonomy of purposes and retrieving user information relevant to a particular purpose just in time in order to assemble and integrate fragmented user model information. This purpose-based user modelling is based on procedural representation of purposes. Each purpose has inputs, outputs and functions, which can be used to retrieve and integrate input information in order to generate desired output. Therefore, when an agent or web-service invokes a purpose for a given user modelling task, the purpose shows how to find relevant user model fragments and how to integrate them by executing a modelling function. This modelling process depends on the resource constraints, e.g. which other services or agents are available at the moment to provide information, what kind of user models they can provide and how much time is available for computation.

Purposes can be organized into generalization or aggregation hierarchies. The set of purposes can thus be viewed at many levels, for example, from general to specific. One purpose can aggregate several sub-purposes. The way sub-purposes are aggregated can be defined by the functions of the super purpose. Some sub-purposes can be called sequentially, i.e. when the super purpose is called, the sub-purposes of this super purpose will be called in one after another as long as there are resources available to continue, in an "anytime" fashion.

A library of purposes forms a repository of clichés, which can be adapted to a new situation. For example, a purpose can be generalized into a super purpose, specialized into (a set of) more specific purposes; shared by several super-purposes; or modified for use in a new domain. Purpose re-use is valuable from a software engineering point of view and critical to the active approach. Another aspect to software re-use is model re-use, which means re-using the result of the computation, i.e. the output of a purpose, as input to other purposes.

The purpose hierarchies within this system architecture are maintained by a set of specialized user modelling web-services associated with each purpose. These agents are networked according to the purpose hierarchies. Each user modelling service asks the next (according to the aggregation or to the generalization dimension in the purpose hierarchy) available UM service to continue the computation needed to achieve the appropriate sub-purpose. In this way web-services and agents subcontract user modelling tasks to the specialized user modelling agents for the purpose, which perform computations upon request and return the results to the requesting agent or service without storing any data. In this way, the computation of user models and the storage of user data in this architecture are fully decentralized. Specialized purpose agents can be reused easily.

In a system with many UM services specialized for a wide range of purposes, an agent- or web-service based application can perform just-in-time user modelling by

calling the appropriate UM service for its particular purpose. Similarly to developing a full ontology of a domain, envisaging all possible purposes for user modelling in all possible contexts is a hard task. However, the decentralization allows multiple designers to gradually create UM services for various purposes and they will become useful immediately. The effort of the designer should focus at the first place on important, reusable purposes that can be either reused directly, or easily adapted.

The decentralization of the user modelling process and focussing it on the particular adaptation purpose at hand is orthogonal to the problem of commonly understandable representation language for user modelling information. There are two approaches to ensure such common understanding: using a centralized ontology for user modelling and mapping different user model representations. The second approach is more interesting since it is also decentralized, and it is a currently active area of research [3, 9].

Our work [5, 6] has demonstrated the use of purposes in a multi-agent portfolio management domain [7]. There are two kinds of agents in the system: personal agents (PA) represent investors who need advice, and expert agents (EA) who provide that advice. There are many purposes for adaptation in this domain. One of the main adaptation purposes for a PA is to find an appropriate EA for a given investor, which is a kind of personalized recommendation. This purpose requires models of the investor and the EA as well as eventually models of other investors and their EAs.

3 Experiments and Results

The goal of the experiments is to demonstrate that this purpose-based user modelling is *feasible* and that there is a methodology of systematic development of purposes through organizing them in hierarchies that allow for reuse. The system was evaluated with personal agents representing simulated investors to show that is provides adapted functionality according to pre-set investors' preferences. The experiments [6] demonstrated the anytime aspect of purposes and also showed that the quality of the decisions improves when more resources and agents are available. These results help to establish a performance baseline for the system and to discover the influence of environment factors on the system and possible measures (e.g. reordering the purposes to take into account such factors as the expected level of deception).

We only ran 40 personal agents in our experiments. The further challenge is scaling up to a larger set of agents. There will be a lot of overheads associated with the multi-agent system, such as finding the agents and communication etc. We expect that the quality of predictions should improve and the speed not worsen as the number of agents grows. This is the case because the more people use the system; the greater the chances are of finding close matches for any particular investor (i.e. neighbours). One potential limit to growth will be communication (influencing network traffic, performance, response time), if the system grows very large. There will always be a trade-off between resource constraints and quality of performance. The anytime algorithms allow an easier solution to this: if time resources are extremely limited, and there is sufficient number of investors, a reasonable performance can be achieved

while executing only one or two simple sub-purposes which are much cheaper and require less communication.

4 Contributions

Our research makes a number of contributions in both active user modelling and software engineering.

- Designing and developing an agent environment for studying decentralized, active user modelling issues.
- Demonstrating how a purpose-based approach can implement the decentralized active user modelling paradigm.
- Providing an extendible approach to active user modelling to allow re-use and adaptation of purposes and anytime algorithms to handle varying modelling constraints.
- Showing an example of how simulation can be used to evaluate an adaptive system.

The purpose based approach and its evaluation is described in detail in [5, 6].

References

1. Browne, D., Totterdell, P., Norman, M.: Adaptive user interfaces. Academic Press Ltd., London, UK, 1990
2. 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, 10 (3-4) (2000), 209-249
3. Giunchiglia, F., Zaihrayeu, I.: Making peer databases interact - a vision for an architecture supporting data coordination. Technical Report # DIT-02-0012. Also to appear in Proc. Cooperative Information Agents (CIA 2002), Madrid, September 2002
4. McCalla, G., Vassileva, J., Greer, J. and Bull, S.: Active Learner Modeling. Proc. ITS2000, Springer LNCS 1839, (2000) 53-62
5. Niu, X., McCalla, G., Vassileva, J., (2003) Purpose-based user modelling in a Multi-agent portfolio management system. Proceedings User Modelling UM03, Johnstown, PA, June 22-26, 2003. Springer Verlag LNCS 2702, 398-402
6. Niu, X., McCalla, G., Vassileva, J.: Purpose-based expert finding in a portfolio management system. Computational Intelligence Journal, Vol. 20, No. 4, (2004) 548-561.
7. Tang, T., Winoto, P. & Niu, X.: Who Can I Trust? Investigating Trust between Users and Agents in a Multi-agent Portfolio Management System. In AAAI-2002 Workshop on Autonomy, Delegation, and Control: From Inter-agent to Groups. Edmonton, Canada, July 28, 2002
8. Vassileva, J., McCalla, G., Greer, J.: Multi-Agent Multi-User Modelling. To appear in User Modelling and User-Adapted Interaction, (2003), ***13:****(1), 179-210.*
9. Hopkins, I., Vassileva J. (to appear) Beyond Keywords and Hierarchies. *Journal of Data and Information Management*, special issue on Distributed Data Management. Available on line at: http://bistrica.usask.ca/madmuc/tagging.htm

Agent Models in Service of Information Dissemination in Inaccessible Cooperative Multiagent Systems

Olayide Olorunleke and Gordon McCalla

ARIES Lab, Department of Computer Science,
University of Saskatchewan, Canada.
{oto033, mccalla}@cs.usask.ca

Abstract. An agent is often required to act based on the information its environment or of other agents. It is possible that such information is not available to the agent even though another agent might have such information. Hence, a difficulty is knowing which of the other agents to contact for a piece of information that is required. We present a solution to this problem by using agent models that estimate the knowledge of other agents in conjunction with the predictive memory model of Bowling et al [1] to pro-actively send gossip to other agents. Empirical evidence observed from experiments suggest that a pro-active approach to information sharing can indeed be effective in making the relevant information available to the agents under certain assumptions.

1 Introduction

The state of the environment at any time *t* is defined to be a vector v_t, the contents of which may be accessed using the dot notation such as $v_t.x$, $v_t.y$, $v_t.z$, etc. With this information available, therefore, each agent can simply use its given policy for acting – which could have been learnt prior using reinforcement learning techniques – as a look-up table to select the optimal action in each perceived state. An environment is said to be *accessible* to an agent if the agent's sensors are always able to detect this current state – as long as the sensors have not failed. When this assumption of accessibility is broken, the environment is said to be *inaccessible*. Inaccessible environments by definition, therefore, make it difficult for the agent to select the optimal action even when it knows the optimal policy for acting in that environment.

One solution to this problem is to use a centralized model server that continuously receives information about the state of the environment (from the agents), aggregates these "snap-shots" into a more global picture and responds to each agent's query from this aggregated global perspective. An example is the PHelpS system [3]. This approach is often criticized, however, for providing a central point of failure for the system as a whole or for not scaling well with the number of agents [4].

Another solution is to use a decentralized approach in which each agent is responsible for managing its own models. In this case, each agent locally maintains information about the environment. The absence of the central model server implies that each agent takes over the responsibility of aggregating the information fragments

that it needs. Among the problems introduced by choosing such a decentralized approach, the one we address in this paper is the following: *how does an agent locate another agent that has the relevant feature of the environment (or another agent) that it needs to make a decision?* In our attempt to answer this question, we make the assumption that the agents are in a cooperative multiagent system. With this assumption made then, we alternatively ask the question as: *how does an agent locate another agent that needs to know a feature of the environment (or a third agent) which this agent already knows?* Asking the latter question suggests a solution that relies on *pro-active informing* rather than the reactive one which the former question suggests. Reliance on gossip and observations alone, however, does not guarantee that the locally maintained models will contain enough up-to-date information to be used for choosing actions in inaccessible environments. For instance, an agent might receive no gossip and make no observations at all and will thus be left with incomplete information that cannot be used for action selection, or if used will result in an inappropriate selection for the current state of the environment. In such situations, Bowling et al [1] proposed the use of a mechanism called a *predictive memory*.

2 The Predictive Memory Model

The predictive memory model [1] maintains the state information – such as the position, velocity, direction, etc – of each object. Due to the possibility of not being able to always observe all of the state features, the predictive model also stores an additional value – for each state feature – that describes the accuracy of the current value assigned to it. In every time slice, the predictive memory approach updates the values of those features of objects that are directly observable from the sensory information obtained from the environment. Additionally, the probabilities attached to these values are set to 1.0. For those values that cannot be obtained from sensory input, the predictive memory approach uses two phases to achieve the update.

The first phase considers those changes that should occur based on the agent's own actions in the last time slice. For example, if the agent's last action was a turn by angle *a*, then it is possible to update the positions of the maintained data elements by correcting for this turn (since they are stored as relative positions).

The second phase in the update of the data stored in memory about the states features applies particularly to mobile objects. The assumption is that mobile objects tend to continue in their direction of motion and thus – even when out of view – there is a short-lived certainty that the objects will continue moving in that direction. Thus, the unseen mobile object's positions are updated using their last-observed velocities and positions. To account for this guess, the probability values attached to these unseen data items are reduced – by multiplying by a decay factor (e.g. 0.9). Additionally, the last observed velocity that was used to update the object's position is also decayed, to reflect the possibility that moving objects eventually slow down.

3 Enhancing the Predictive Memory via Gossip

To understand the processes involved in our use of gossip in this paper, consider the following definitions:

Definition 1:

if $v_t^A.x$ refers to the value assigned to feature *x* of the state vector at time *t* by agent *A*, and $C_{v_t^A.x}$ indicates how confident *A* is in this value, then we say that *A* is more confident than *B* about *x* at time *t* iff $C_{v_t^A.x} > C_{v_t^B.x}$.

Definition 2:

If agents *A* and *B* use the predictive memory approach, then they each have a subset (possibly empty) of features of the state for which they are more confident than the other. That is,

$$G_A^B = \{(x, v_t^A.x, C_{v_t^A.x}) \mid C_{v_t^A.x} > C_{v_t^B.x}\} \text{ and } G_B^A = \{(x, v_t^B.x, C_{v_t^B.x}) \mid C_{v_t^B.x} > C_{v_t^A.x}\}$$

Definition 3:

When *A* gossips with *B*, the message $I_{A->B}$ that is sent by *A* to *B* must be a subset of G_A^B, i.e., $I_{A->B} \subseteq G_A^B$. If $|G_A^B| = 0$ then no message is sent.

Definition 4:

When *B* receives gossip from *A* about any feature *x*, it updates this feature in its predictive memory using the following:

$$C_{v_{t+k}^A.x} = decay^k * C_{v_t^A.x}; if\ C_{v_{t+k}^A.x} > C_{v_{t+k}^B.x}\ then\ v_{t+k}^B.x = v_t^A.x\ and\ C_{v_{t+k}^B.x} = C_{v_{t+k}^A.x}$$

Note that by this definition we assume that the message sent by *A* at time *t* will be received by *B* at *t+k* (where it took *k* cycles for delivery).

Proposition 1: If *A* gossips with *B* about feature *x*, then *B* can do no worse if it accepts *A*'s gossip into its predictive memory than if *A* did not gossip at all, and thus gossip can be used to enhance the predictive memory approach.

Proof. To prove this proposition, we consider two possible cases.

Case 1: When the feature value passed by *A* is accurate. In this case, *B* basically accepts accurate information or keeps its old beliefs. Either way, this is no worse than if *A* did not gossip at all.

Case 2: When the feature value passed is inaccurate. By definition, confidence values are only set to 1 when features are directly observed. If sensors are assumed to be perfect then a confidence value of 1 can be taken as a sign of accuracy. The case in which inaccuracy is evident then is when the confidence values are less than 1. Recall from definitions 2 and 3 that the contents of the messages are only drawn from those in which the gossiper's confidence is greater than the recipients'. This is only possible if the gossiper observed the feature at a certain time interval after the recipient had observed the same feature, thus making the gossiper's estimate more accurate. So the recipient (*B*) either changes to more accurate information or sticks with its original value. Thus, gossip can indeed be used to enhance the predictive memory approach. □

Despite the promise of gossip being useful in inaccessible environments, however, a few problems arise that must be tackled. The first of these arises from definition 2. That is, *A* is required to know both $C_{v_t^A.x}$ and $C_{v_t^B.x}$ in order to select G_A^B. Knowing $C_{v_t^A.x}$ is trivial since it is maintained locally by *A*. However, knowing $C_{v_t^B.x}$ presents a challenge for A because *B* maintains such knowledge locally. The second problem arises in definition 3. That is, how should the subset of G_A^B that is sent by *A* to *B* be determined?. The third problem arises from problems 1 and 2. That is, since *A* might not know *B*'s confidence in the value of feature *x* for certain, if *A* makes an estimate of this value and is wrong, then *x* would have made it into G_A^B even though *B* might actually be more confident than A about *x.* This in turn makes it possible that if *A* selects *x* to be in $I_{A->B}$ (i.e., *A* gossips with *B* about *x*), then by definition 4, *B* would reject *A*'s value for *x*. Although this error by *A* does not affect the predictive approach negatively, it results in a waste of communication bandwidth, and robs *B* of the opportunity to have received a value it actually needed.

To solve the first of these problems, we use agent models such that *A* keeps a model of *B* that only contains estimates of *B*'s confidence in each feature. Since we are unable to guarantee that the contents of the agent models are accurate, we propose to reduce the amount of wasted bandwidth by proposing the following heuristic.

Heuristic 1: *A* should also maintain a threshold for the modeled values such that it would only include *x* in G_A^B if $C_{v_t^A.x} > C_{v_t^B.x}$ and $C_{v_t^B.x} < threshold$. Note that in this case $C_{v_t^B.x}$ is obtained from *A*'s model of *B*.

4 Validation of the Gossip-Based Enhancements

Although we have formally proved that gossip can be useful in enhancing the predictive memory approach, we also verified this by running an experiment in the RoboCup domain in which a comparison is made between the time it takes to complete a task without proactive gossip and the time it takes with proactive gossip. The use of proactive gossip resulted in the task being completed in 70 cycles (down from 150 cycles without proactive gossip). We also wanted to confirm that the use of heuristic 1 could result in a reduction in the amount of bandwidth that is wasted by gossiping. We used a threshold value of 0.6 as described in heuristic 1. A total of only 15 extraneous messages were sent by the time the task was completed when heuristic 1 was used (down from 35 without heuristic 1).

5 Discussion and Conclusion

For this approach to managing and updating the locally held models to be transferable to other domains where models are distributed across agents, the assumptions we have made (which hold in the RoboCup domain) have to hold (or be made to hold where possible) in these other domains. The first of these is that the agents can observe each other. From this observation, it is possible to extract information that can be used to infer what an agent knows or does not know. For example, in an e-learning system, this could translate to allowing user agents to observe interactions that take place between each other. The second implicit assumption is the altruistic nature of the agents involved such that they are always willing to fill-in gaps in each other's knowledge when this is detected. This stresses the need for the research on persuasion and motivation already being pursued in [2], etc.; the results of which will help in ensuring that the agents or users are motivated to help each other. In this paper, agent models have been employed that contain estimates of what each agent knows. We have shown how these agent models can be used in decision making on when to gossip with other agents and in selecting the information that is included in the gossip. We have also shown a heuristic that can be used with these agent models to reduce that amount of bandwidth that is wasted when agents gossip with each other. We have not addressed the possibility that even though the agents do not lie about the information that is passed via gossip, such information can still be wrong (for instance, when sensors fail). In our current ongoing work we are studying various additional strategies [5] that can be used by the recipient to avoid believing the incorrect information it receives via gossip.

References

1. Bowling, M., Stone, P., and Veloso, M. (1996) Predictive Memory for an Inaccessible Environment. In Proceedings of the IROS-96 Workshop on RoboCup, pp. 28–34, Osaka, Japan, November 1996.
2. Cialdini R. (2001) Influence: The Science of Persuasion. Scientific American, pp. 76-81, February, 2001.
3. Collins, J., Greer, J., Kumar, V., McCalla, G., Meagher, P. and Tkach, R. (1997) Inspectable User Models for Just in Time Workplace Training. In A. Jameson, C. Paris, C. Tasso (eds.) User Modelling, *Proceedings of the 6th Conference on User Modelling, UM97,* Springer Wien, New York, pp. 327-337, 1997.
4. Fink, J. and Kobsa, A. (2000) A Review and Analysis of Commercial User Modelling Servers for Personalization on the World Wide Web. In User Modelling and User-Adapted Interaction, Special Issue on Deployed User Modelling, pp. 209-249, 2000.
5. Olorunleke, O. and McCalla, G. (2004) The Study of Delusion in Multiagent Systems. In Proceedings of Workshop on Modelling Other Agents from Observations (MOO 2004), pp. 33-40, New York, 2004.

Sharing Models of Sellers amongst Buying Agents in Electronic Marketplaces

Kevin Regan[1], Thomas Tran[2], Robin Cohen[1]

[1] School of Computer Science, University of Waterloo
[2] SITE, University of Ottawa

1 Introduction

In this paper, we demonstrate the value of decentralized models of sellers in electronic marketplaces, as the basis for purchasing decisions from buyers. We discuss how buying agents can model the reputation of sellers in order to make effective purchases and how these agents can also take advantage of reputation ratings provided by other buying agents in the marketplace, once it is established that each buyer will be independently modelling the sellers. We outline the methods required to make use of reputation ratings of sellers provided by other buyers, including adjustments for possibly different subjective scales and for possible deception in reporting the reputation ratings. In all, we have a community of adaptive applications effectively sharing information about possible sellers.

2 Model

Our model builds on that that of Tran and Cohen [1], described briefly below.

Definition 1. *Given a set S of sellers, we denote the reputation of a seller $s \in S$ as seen by a buyer b as $r_s^b \in (-1, 1)$.*

Definition 2. *$f : G \times P \times S \rightarrow R$ is the estimated value function used by a buyer to assess the value of a good $g \in G$ given the price $p \in P$ and seller $s \in S$. We generally denote the estimated value function for a buyer b as $f^b(\cdot)$.*

We use a reputation threshold Θ and a disreputation threshold θ to partition the set of sellers. Sellers for whom $r^b > \Theta$ are deemed reputable (R). Sellers for whom $r^b < \theta$ are deemed disreputable (DR), while the rest of the sellers are put into the set (?)[3] which the seller is unsure of. We can formally express this as follows

$$\forall\, s \in S \;\; s \in \left\{ S_R^b \text{ if } r_s^b > \Theta;\, S_{DR}^b \text{ if } r_s^b < \theta;\, S_?^b \text{ otherwise} \right\} \tag{1}$$

The reputation of a seller is adjusted based on the resulting value of a transaction v^b and a buyer's satisfaction threshold ϑ^b. When $v^b \geq \vartheta^b$, the buyer is

[3] Tran and Cohen describe this set as those who are neither reputable nor disreputable

satisfied and the seller's reputation r_s^b is increased by $\mu(1 - r_s^b)$. When $v^b < \vartheta^b$, the buyer is unsatisfied and the seller's reputation is decreased by $\nu(1 - r_s^b)$.

The buyer choses the seller with the highest estimated value $f(\cdot)$ from among the reputable sellers. The potential sellers who have been deemed disreputable are never purchased from and the sellers a buyer is unsure of are occasionally used to buy goods from. The buyer selects a potential seller from the set $S_?^b \cup S_R^b$ with some small probability ρ in order to explore new sellers.

We move beyond the model presented by Tran and Cohen [1] to provide an approach using seller ratings provided by other buyers.

Consider the situation after a buyer b has made a request for a good and received bids from a set S^p of potential sellers. In some situations it may be beneficial for the buyer to ask a set of other buyers about the potential sellers. For instance, when a buyer chooses a seller for the first time, or simply does not have much information about as seller it should consult other buyers. We refer to other buyers in this role as *advisors*. For each advisor $a \in A \subseteq B$ our buyer will maintain a reputation r_a and partitions A_R, $A_?$, and A_{DR} in the same manner as seller information is maintained.

$$\forall\, a \in A \quad a \in \begin{cases} A_R^b & \text{if } r_a^b > \Theta' \\ A_{DR}^b & \text{if } r_a^b < \theta' \\ A_?^b & \text{otherwise} \end{cases} \tag{2}$$

The reputation of an advisor will be updated following a purchase when the buyer will either be satisfied or unsatisfied with the true quality of the good based on our satisfaction threshold ϑ. We essentially adjust the reputation of each advisor based on whether they were right or wrong about the seller. There is an increase: if we were satisfied when the prediction was reputable, or if we were dissatisfied when the prediction was disreputable. There is a decrease if the satisfaction and reputability are at odds.

We use the constant factors α and β to define the amount of the reputation adjustment. The adjusted reputation of an advisor a after an increase is defined as

$$r_a^b \leftarrow r_a^b + \alpha(1 - r_a^b) \text{ if } r_a^b \geq 0;\ r_a^b \leftarrow r_a^b + \alpha(1 + r_a^b) \text{ if } r_a^b < 0 \tag{3}$$

while the adjusted reputation of our advisor after a decrease is defined as

$$r_a^b \leftarrow r_a^b + \beta(1 - r_a^b) \text{ if } r_a^b \geq 0;\ r_a^b \leftarrow r_a^b + \beta(1 + r_a^b) \text{ if } r_a^b < 0 \tag{4}$$

In the preceding formulae α and β are positive and negative factors respectively and are chosen according to the preferences of each individual buyer.

After the adjustment of an advisor's reputation, the advisors can be repartitioned into reputable, unsure and disreputable sets using equation (2). This model of advisor reputation is used to decide which advisors to consult and how to interpret their feedback. For instance, a buying agent will avoid returning to advisors who have been moved into the disreputable set after an adjustment and will only ask agents in the set of non-disreputable advisors (i.e. those in the set

$A^b_R \cup A^b_?$) about a set S^a of sellers. The set S^a is composed of all the potential sellers the buyer is unsure about as well as a set $S^a_!$ of sellers which the buyer already knows about which is taken from $S^b_R \cup S^b_{DR}$. $S^a_!$ will allow our buyer to assess how each advisor's standards differ and to adjust in order to correct for these diffferences.

The advisor responses are combined to form a temporary reputation r^A_s for each seller. This new reputation is used to construct a set of reputable potential sellers (as in equation 1) from which the buyer can make a more informed purchase decision. The way in which the advisor responses are combined must take into account the differing subjective standards used by each advisor to assess reputation as well as the possibility of the advisor being untruthful or inaccurate.

Definition 3. *For each advisor a that responds to the buyer b 's request and seller* $s \in S^a_!$*, we calculate the reputation error* $\epsilon^a_s = r^a_s - r^b_s$

Definition 4. *We denote the mean and standard deviation of the reputation error over a set of sellers as* $\bar{\epsilon}^a$ *and* σ^a *respectively.*

We can adjust for systematic differences (ie. σ^a is small) using the equation: $\forall\ s \in S^a,\ \ r^a_s \leftarrow r^a_s - \bar{\epsilon}^a$

Our buyer will use the reputation held for each advisor to mitigate the effects of deceptive or inaccurate reputations given by an advisor. To avoid confusion between these two notions of reputation, we will occasionally refer to the reputation an advisor has about a seller as a prediction, since when this is information is passed on to the buyer and used as indirect reputation the advisors are, in a sense, making a prediction about the outcome of the buyer's purchase.

The responses from each of the advisors are combined so that the effect of dishonest sellers is minimized. However, each advisor is assumed to be honest until we find sufficient evidence of deception. It should be noted that we do not adopt the approach of weighing an advisor's predictions by the advisor's reputation ($r^b_a \cdot r^a_s$) that has been used by others [3]. The argument for our approach is that a until an advisor is no longer reputable, it is beneficial to fully consider their prediction (and not dilute it by some fractional weight).

We lessen the impact of dishonest sellers by maintaining reputations for each advisor and only use the predictions of the reputable advisors. We begin by finding the average over all the reputable advisors for each reputable seller.

Definition 5. *Given a seller s and a set of reputable advisors* $A^b_R \subseteq A$*, we denote the average prediction about s over all* $a \in A^b_R$ *as* $\bar{r}^A_s$*.*

An advisor with a high reputation who decides to lie about a particular seller can still have a large impact. This is particularly relevant since we assume all advisors are reputable until proven otherwise. To lessen the impact of reputable dishonest advisors we can choose to ignore predictions that are significantly different from that of the other reputable advisors. As a measure of significant difference we use the standard deviation of the prediction given by the reputable advisors,

which we denote σ_s. We then adjust seller reputation using the following rule: $r_s^A \leftarrow$ avg r_s^a over $a \in A_R^b$ where $|r_s^a - \bar{r}_s^A| < \sigma_s$.

It should be noted that after a purchase a buyer's reputation for *all* of the advisors contacted is updated. An advisor's reputation can increase even if it was ignored when the seller was being chosen. In this way an advisor who fell below the reputable threshold can be redeemed.

3 Example

In this example we have only two potential sellers (s_r and s_{dr}) among whom our buyer b must decide to buy a good. The seller s_r has never deceived a customer, while s_{dr} has lied to customers. However, our buyer b, has no experience with either seller and seeks help from a set of advisors (a_1, a_2, a_3, a_4) with respective reputations (0.1, 0.5, 0.6, 04). Our advisors a_1, a_2, a_3, a_4 provide respective seller reputations for s_r of $(-0.25, -0.6, -0.7, 0.2)$, and the respective seller reputations for s_{dr} of ($1.0, 1.0, -1.0, -0.5$). Our buyer fixes the advisor reputation thresholds at $\Theta' = 0.20$ and $\theta' = -0.20$ resulting in a_1 being selected from the set $A_?^b$, while a_2, a_3 and a_4 are selected from the set A_R^b. For the purposes of our example, a_1 turns out to be deceptive and provides deliberately inaccurate reputation information. The advisor a_2 is truthful, but has had good non-representative experiences with s_{dr} and provides an overly high reputation for this seller. Both a_2 and a_3 have high standards and this lowers the reputations they provide for each seller accordingly. The advisor a_4 iss truthful and has similar standards to our buyer.

Now, our buyer b receives a reputation for s_r, s_{dr} and $s_! \in S_!$ from each advisor and if b were to simply average the reputations for s_r and s_{dr} without the methods developed to account for deception or differing standards, the result would be a reputation of -0.34 for s_r and 0.13 for s_{dr}. Now, let's say that b partitions sellers using: $\Theta = 0.20$ and $\theta = -0.20$ (as in equation 1), since $-0.33 < \theta$, s_r would be added to the set of disreputable sellers and since 0.13 is between θ and Θ, s_{dr} would be added to the set of sellers our buyer is unsure about.

The first step towards extracting accurate reputation information from our advisors is to account for any systematic bias. Our buyer finds the average difference between the reputation it holds and the reputation the advisor holds for each common seller $s_i' \in S_!$ [4] In the case of a_2 and a_3, our buyer finds a difference of $\bar{\epsilon} = -1$ and a low σ indicating that our advisors consistently under-appreciate sellers by about -1. The buyer will adjust the reputations given by a_2 and a_3 by $-\bar{\epsilon}$. In our example 1 will be added to the reputations given by a_2 and a_3 and the average reputation for s_r and s_{dr} rises to 0.16 and 0.38 respectively[5].

The second step is to ignore any reputation information from advisors that our buyer is unsure about. Here, the buyer ignores the deceptively low reputation that a_1 provided for s_r and the deceptively high reputation that a_1 provided for

[4] The reputation ratings for each s_i' held by the buyer and advisor are omitted here

[5] After adjustment a reputation greater than one will be normalized to one

s_{dr} resulting in s_r's reputation rising to 0.30 and s_{dr}'s reputation dropping to 0.17. The seller s_r is now in our buyer's reputable set, however our buyer is still unsure about s_{dr} due to the inaccurate high reputation given by the truthful advisor a_2.

The third and last step calculates the standard deviation of the set of reputations provided by reputable advisors and eliminates any reputation given by these reputable advisors that deviates from the average by more than one standard deviation. The unrepresentative high reputation provided for s_{dr} by a_2 is eliminated and the resulting average reputation for s_{dr} drops to -0.25 moving s_{dr} into the set of disreputable sellers. In our example the methods developed in this paper have successfully limited the effect of differing standards, and deceptive or inaccurate advisors. The buyer selects s_r and the reputation of the advisors is adjusted, depending on whether the buyer is satisfied with the purchase and the predictions of the advisors. After the interpretation phase our advisors a_1, a_2, a_3, and a_4 have given a reputation of (-0.25, 0.40, 0.30, 0.20) for s_r which predict s_r falling into the following respective sets (disreputable, reputable, reputable, reputable). Suppose that the buyer is satisfied with its purchase and our constant increase and decrease factors α and β are set to 0.2 and 0.4 respectively. This will result in the reputation of a_1 being decreased by $\beta \cdot (1 - r^b_{a_1}) = 0.36$, which will move a_1 into the set of disreputable advisors. The reputations of a_2, a_3 and a_4 are increased to (0.52, 0.60, 0.68) respectively.

4 Discussion

This approach contrast with that of Yu and Singh [2], which appeals to advice from witnesses but does not account for differing standards in determining the reputation of sellers. It also compares to the Sporos system [3] which combines the ratings from a group of users, but does not consider how to find these other agents or how to address subjectivity.

Our research presents strategies for making decisions about sellers based on models of advisors' deceptiveness and subjectivity. This is a decentralized approach to representing sellers within the marketplace, where data harvested in one context is useful for adaptation in another, with each individual buyer managing its own processing. The modeling of trust between users and coalition formation based on trust are relevant issues within our framework.

References

1. T. Tran and R. Cohen. Improving user satisfaction in agent-based electronic marketplaces by reputation modelling and adjustable product quality. In *AAMAS04*, New York, USA, July 2004.
2. B. Yu and M. Singh. Detecting deception in reputation management. In *AAMAS03*, pages 73–80, 2003.
3. G. Zacharia, A. Moukas, and P. Maes. Collaborative reputation mechanisms in electronic marketplaces. In *32nd Hawaii International Conference on System Sciences*, 1999.

Towards a Framework for Distributed User Modelling for Ubiquitous Computing

Marcus Specht, Andreas Lorenz, and Andreas Zimmermann

Fraunhofer Institute for Applied Information Technology
Schloss Birlinghoven
53754 Sankt Augustin, Germany
{Marcus.Specht, Andreas.Lorenz, Andreas.Zimmermann}@fit.fraunhofer.de

Abstract. This paper introduces a framework for distributed user modelling in ubiquitous computing. Regarding the requirements in ubiquitous computing, communication details are not fixed per se or even unknown during the system specification phase. Hiding the technology within specialized components allows for unique migration of information between the components and releases system developers from considering specific properties and protocols. Supporting efficient development of applications, the distributed components are able to react both to their environment and to messages received from neighbouring components.

Introduction

In the vision of ubiquitous computing the technology becomes invisible to the user and will be embedded in the objects of our daily life. The user will be surrounded with capabilities for accessing information and services everywhere with many different information devices. These information devices have a direct contact to each other in order to offer common services. The user has *one personal information space* independent of devices and the system manages the information spaces of its users. To have access to this personal information space, distributed components on heterogonous platforms need to exchange information with one another. On the one hand users will use personal devices they carry with them (like mobile phones), on the other hand they will access stationary access points connected to the network. Future applications therefore have to support the information exchange between those "user terminals" and their available sensor networks like location tracking facilities built into a mobile phone and stationary sensors hooked up to the network like environmental sensors. Furthermore for each single user, her actual task and current situation an application has to select the most appropriate device for interaction.

In this sense for future application development for personalized and contextualized applications we expect centralized design-approaches to be confronted with a variety of clients running on heterogeneous devices with different properties, such as personal or wearable devices of the users, embedded technology in displays or printers, and everyday-objects like keys or coffee-machines. Following classical centralized approaches of adaptive information selection and presentation [1], passing

all data from clients to servers for analyzing and centralized decision on adaptation means will cause high network traffic and computational requirements on servers. Furthermore, central user-modelling servers [2] holding all information from registered users will not be applicable for two other reasons: On the one hand, every device of the mobile users will not have permanent access to the server, and on the other hand, the local system will not constantly need all information about the user to make a local decision like contextualization to a current environmental change. Each local component needs to use local information and must be able to integrate in an integrated *representation of knowledge* about the user. In our vision, components for user model acquisition and user model application are equally distributed with the user model itself.

Distributed User Modelling

The provision of personalized information services becomes a complex task in open and distributed environments of mobile users. Mobile applications, in particular in ubiquitous environments, rely on a network of sensors placed within the physical environment and watching indicators for changing situations. On the other hand, the actuators are specialized software-components that process the delivered data or display information snippets on a particular device. The aim of *distributed systems* is to spread the application logic among different parts hosted on different physical devices. For the mobile user, the devices will continuously connect to local networks and therefore will have access to all information available in this network. A centralized solution fails because of its inability to cope with a high degree of change, which requires that the solution is both robust to disruption and self-configurable. For *distributed user modelling* approaches this implies that monolithic user modelling is replaced by distributed user model fragments [3, 6]. The information about the user, i.e. the current state of the user-model, will be merged with all information that can be requested from components reachable in the current context.

In resent research in smart *sensor-networks*, sensors build ad-hoc *self-organizing networks* and deliver requested information on demand. In difference to such sensor-networks, distributed modelling components actually receive pre-processed data from virtual components instead of direct measuring the physical environment. Providing a framework will enable all applications on the devices to check into the network and to make use of the available user-related information. Although this information will just be a cut-out of all potentially known knowledge, it will reflect the current environment by including all relevant information at the current location. In the next section we will illustrate our approach for supporting the information-exchange between ad-hoc networked components.

Framework for Distributed User Modelling

Although there exist approaches for applying mobile agents considered to migrate between devices and always stay with the user, we will focus on supporting migration of information in a network of distributed components. By introducing brokering

components on the local host we create unique communication interfaces for all local components. From our previous work [5, 8] we propose to have components for sensoring, modelling, controlling and actuating distributed on different devices. The basic underlying cooperation-approach between those components is *cooperation by information-exchange*. For knowledge-exchange and command-delivery, the components share local message-boards on their hosting devices. The message-boards are managed by specific information-brokers: Locally, the broker provides access to a message-board whereas the information exchange between devices is based on message-sending between the brokers. An important difference to existing solutions is the level of integration of user information as we focus on an information integration of personal and contextual information about a user in the current context and not a general integration which is also critical from a privacy point of view.

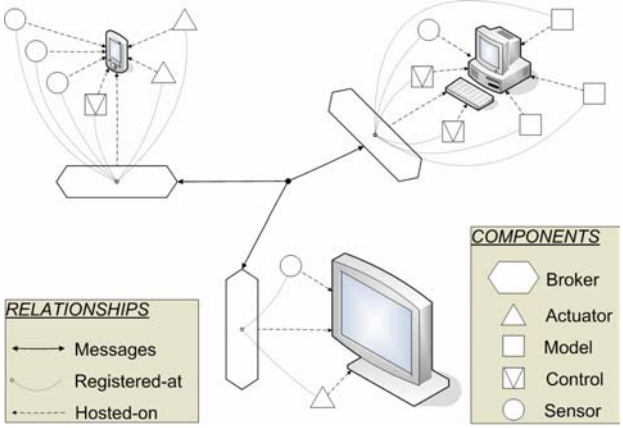

Fig. 1 Distributed User Modelling Components

As illustrated in Fig. 1, different components host physically on different devices. Each component registers at the local board based on defined check-in/check-out mechanisms, announcing what information they provide and what information they request. Potentially, each component can provide any number of attributes and in turn listen to any number of incoming events. The broker manages the list of registered components on the device and cross-links their names in two maps:

1. *Map of local information providers:* The name of the component is mapped to the attributes it provides.
2. *Map of local information listeners:* The attributes are mapped to the names of components who have registered as listeners.

Mapping the components by their names introduces the requirement of unique attributes, whereas it is not required to have one-to-one relationships between attribute names and information providers. Currently, the order of check-in messages defines the active component: The last registered information provider for an attribute will be mapped to be its information provider, overwriting the former one. For sharing information between applications, we will need to install any kind of name-service, and a common user model exchange language (such as *UserML* [4]) supporting the communication between different user adaptive systems.

The Information Flow

When any component hosted on a device sent a message to the local broker, it will be forwarded to all other brokers. For example, the broker at the PDA in Fig. 2 forwards the message to the broker at the desktop via WLAN as well as to the smart-board broker via Bluetooth. The receiving brokers will forward the messages to local receivers, if any registered component was interested in this data.

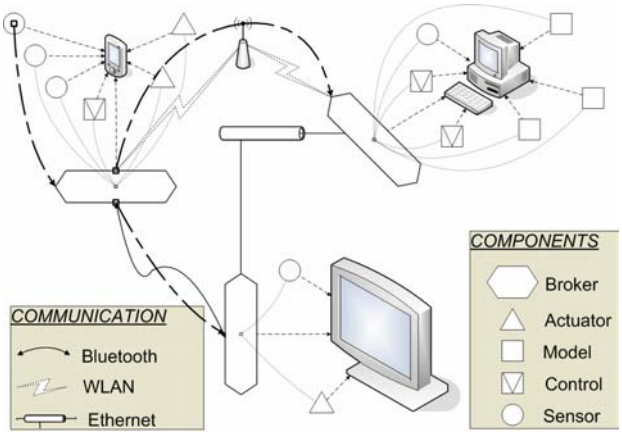

Fig. 2 The Information Flow

Communication among the components usually isn't just a random exchange of messages. Two standard negotiation protocols manage the message flow between the components:

- *Question-Reply*: At check-in, each component has announced what information it delivers. In question-reply protocol, the local component requests missing information from the broker hosting on the local device. The broker broadcasts the request towards all surrounding brokers reachable in the current technical context. Each receiver checks the map of local information providers for a registered component. If such a component was found, the broker forwards the request locally. After receiving the answer, the broker returns the message to the requesting broker, which in turns replies to the question of the local component.
- *Subscribe-Inform*: At check-in, each component has registered as a listener to all information it needs to be informed. If one of the components fires an event, the event-message is sent to its local broker, who forwards the message to all surrounding brokers. Each receiver checks the map of local information listeners and forwards the event-message to each registered component.

Conclusions and Future Work

In this paper we introduced a framework for future user-adaptive application development in ubiquitous computing. We have illustrated our approach to implement a communication-platform hosting distributed components. In this approach, the

knowledge about the user, i.e. the current state of the user-model, will be assembled from many entities reachable in the current context. The devices will continuously connect to local networks and therefore will have access to all information available in this network. Providing a framework will enable all applications on the devices to check into the network and make use of the available user-related information. Usually, the cut-out of potentially available knowledge will reflect the current environment by including all relevant information at the current location automatically. In the current state of work, we specified the platform, defined and implemented messages for check-in/-out, information-request and response, registration of listeners and event-firing, and exceptions. In realisation of our approach, we will implement cooperating agents as active components hosting on the devices and using the communication-framework [7]. As illustrated in this paper, instead of a one-to-one relationship between the user and an agent, sets of agents will be implemented for distributed user-modelling, user-model acquisition and user-model-application.

References

[1] P. Brusilovsky. Methods and techniques of adaptive hypermedia. *User Modelling and User-Adapted Interaction*, 6(2-3):87-129, 1996.

[2] J. Fink and A. Kobsa. A review and analysis of commercial user modelling servers for personalization on the world wide web. *User Modelling and User-Adapted Interaction*, 10(2-3):209–249, 2000.

[3] V. Chepegin, L. Aroyo, P. De Bra, and D. Heckmann. User Modelling for Modular Adaptive Hypermedia. Workshop on Semantic Web for E-Learning, Eindhoven, The Netherlands, pp. 366-371, 2004.

[4] D, Heckmann and A. Krüger. A User Modelling Markup Language (UserML) for Ubiquitous Computing. In P. Brusilowsky, A. Corbett, and F. de Rosis (eds), *Proceedings of the 9th international conference on user modelling*, pp. 403–407, Johnstown, USA, Springer-Verlag, 2003.

[5] A. Zimmermann and A. Lorenz. Augmentation of Content with Context Meta-Data. Workshop on Managing Context Information in Mobile and Pervasive Environments, Ayia Napa, Cyprus, 2005

[6] J. Vassileva. Distributed User Modelling for Universal Information Access. In Stephanidis C. (ed.), *Universal Access in Human - Computer Interaction*, Lawrence Erlbaum: Mahwah, N.J., 122-126, 2001.

[7] A. Lorenz. Agent-Specification for Distributed User Modelling for Ubiquitous Computing, Workshop on Decentralized, Agent Based and Social Approaches to User Modelling, Edinburgh, 2005, to appear

[8] A. Zimmermann and A. Lorenz. Creating Audio-Augmented Environments. *Journal of Pervasive Computing and Communication*, 1(1):31-42, 2005.

AUTHOR INDEX