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.

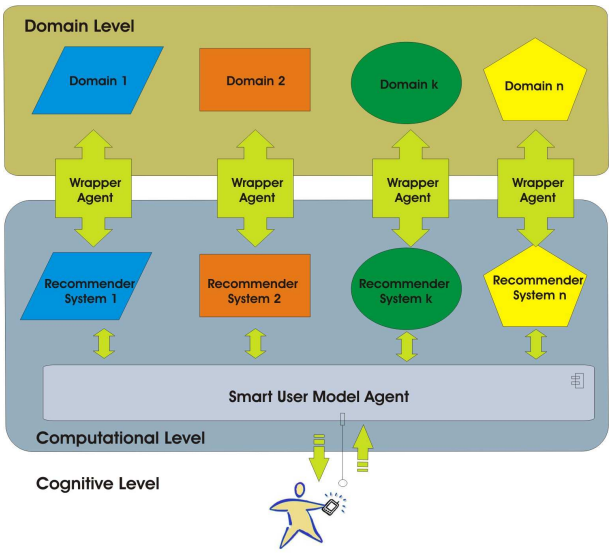

Fig. 1. The Smart User Model in Ambient Intelligence

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, $U^F = \left[(a_p^F, v_p^F)_p\right]$ 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.

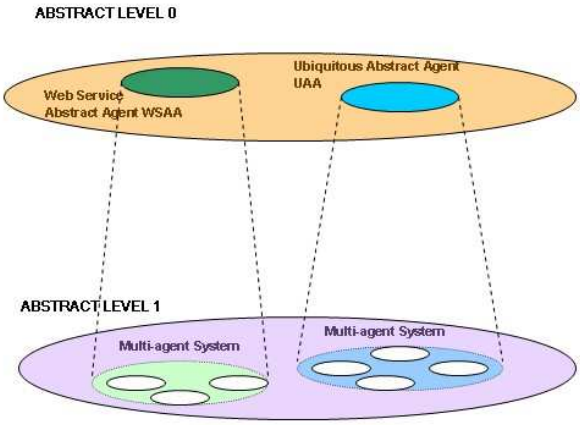

Fig. 2. Two-level Abstract Architecture for the Smart User Model

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

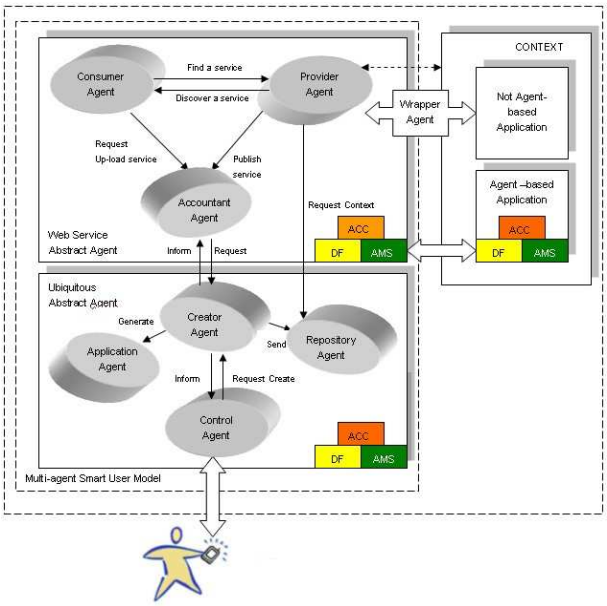

Fig. 3. Multi-agent System Architecture for the Smart User Model

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 [10,11,12,13]. 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 [16,17,18] 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 Recommender 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 Approach 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-Aware 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