A Recommender Agent Development

Juan C. Ramirez[1], Darnes Vilariño[1], Fabiola López[2],

[1] Faculty of Computer Science, Benemerita Univesiad Autonoma de Puebla (BUAP), Av. San Claudio s/n esq. 14 sur, Ciudad Universitaria, Puebla, México.
[2] Direccion General de Innovacion Educativa, Benemerita Univesiad Autonoma de Puebla (BUAP), Av. San Claudio s/n esq. 22 sur, Ciudad Universitaria, Puebla, México.
jcramirezmx@gmail.com, darnes@cs.buap.mx, fabiola.lopez@siu.buap.mx

Abstract. We present the development of different modules from a recommender agent, an update of the algorithm for rating services; and besides, some learning rules needed to properly recommend services. Recommender agents have been a great help for users in Web's searching processes and information retrieval. This paper's proposal to show the development advances in a recommender agent who links agent theory, recommender systems, and virtual worlds, which becomes a highly innovative subject. Programming of diverse agent modules has been developed on Linden Language Script (LSL), Visual C#, and MySQL. Currently we have a prototype in our server.

Keywords: Recommender Agent, user profile, learning rules, services.

1 Introduction

Nowadays education is leaving behind traditional teaching methods and adopting new learning techniques thus, new concepts arise as e-Learning, virtual learning environments, among others. E-Learning is the educational process placed through mechanisms supported by Internet technologies using audio, video, text, and graphics transmissions [1].

There are places around the world where learning techniques based on virtual worlds are highly used. Teachers and students coexisting in the same virtual environment (like a classroom), but all of them in different locations is a very interesting idea, so many projects are being developed to implement this teaching technique in near future.

There are several approaches trying to combine virtual worlds environments and e-learning systems, an example of this is Sloodle [2], which is a fusion of Secondlife environment and Moodle platform.

Moreover, Retrieval Information area has taken great boom among researchers by the need to develop new software techniques for helping users in search processes. Usually a user loses a lot of time finding interesting information looking for quantity or quality of it.

A solution for this problem is using recommender agents. The function of these agents is to search information related to a user and, as the name involves, recommend to him.

Recommender Systems try to be one step forward from traditional information retrieve, which works by using keywords to find information about a topic through the well-known search engines (google, lycos, yahoo, etc). As the name indicates the recommender systems recommend or suggest to the users the items or products based on their preferences, they are used by Web sites, like electronic commerce, as marketing tools in order to increase the sells presenting to the user those products that user wishes (or may wish) to buy. Thus, we can know what customers like or dislike [3].

Nowadays many websites offer recommendations to their users, these sites are mainly focused to online sells, and of course, the recommendation systems are focused to that area too. An example of this is the E-cousal system, which is based on catalogue sells and supported by a web page and an application functioning as a multi-agent system. [4].

In virtual worlds, users are represented by avatars and they can interact among each other in a 3D world. In those environments there are different services offered to them, with a drawback: the larger a virtual world becomes, the harder it is to find new services or activities added to it (just like the real world: the larger a city becomes, the harder it is to find certain places), these services could increase in a chaotic and uncontrolled way, causing serious limitations for their management, organization and retrieving.

The paper's proposal is to show the advances in the developing of an agent in a virtual world; showing how is built the user profile and the implementation of different modules of the system.

In Section 2 we describe architecture of the agents; in Section 3 we talk about the services and how they are managed; in Section 4 we show the implementation of 4 modules: database, the user profile, the algorithm for rating recommended services and some learning rules; in section 5 we describe some results and finally in section 6 the conclusions and future work.

2 Architecture

The main components of the recommender agent are shown in Figure 1 [5]. Components are listed on rectangles whereas the exchange of information is listed with arrows unidirectional or bidirectional depending on the function. All of these run under OpenSim platform, which is an open source virtual world simulator [6]. In this work we explain in detail how we build the user profile, the Database implementation, and part of the learning techniques, an improvement of the algorithm for rating services and how the services used by the recommender agent are managed.

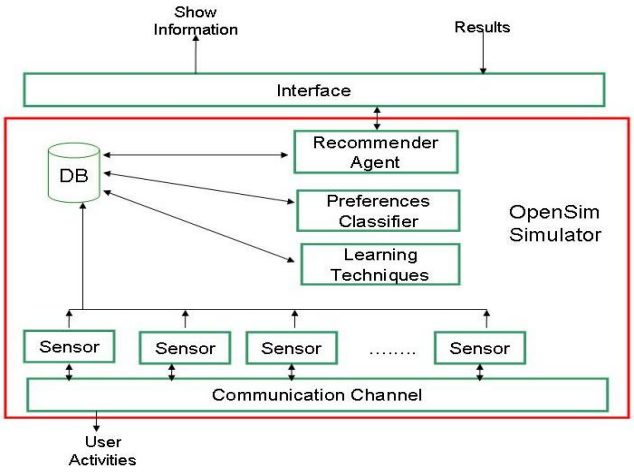

Fig 1. System Architecture

3 Services Module

The services in a virtual world could be varied, from recreational activities to debates or discussions about a specific topic. Services are divided into these categories: Introductory courses, Conferences and Interactive talks, Classes, School courses, Recreational courses, and Parties, luncheons and events.

At this moment the creator of a service is responsible to introduce the service information in the database, filling the required information in a Web page. The data required are: service description, title, keywords and type of service among others. The recommender agent needs this information in order to recommend services. The options for the service creator are shown in Figure 2.

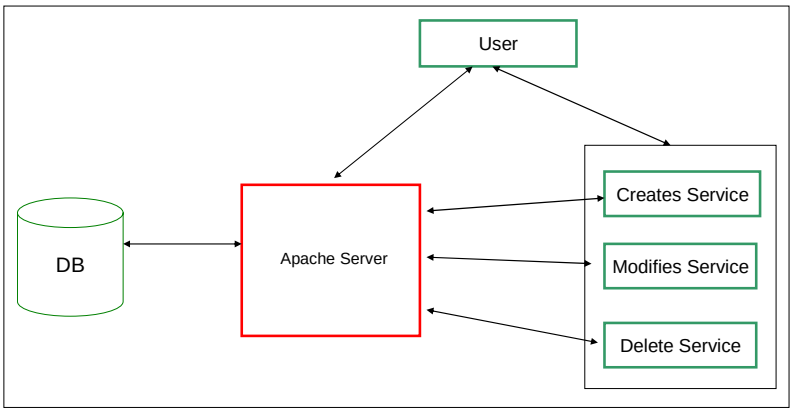

Fig 2. Services Registration Architecture

The user access to their services through a web page bases on Apache server, it allows to create, modify, or delete a service related to that user. When a user creates a service the data required is inserted as a new row in the database and it's related to the user, if the user modifies a service, previously created, the information is updated on the database. The user also can delete a service, in this case the row is not deleted from the database, and the service is only marked as disable.

Using this page, users can offer their services and they can be recommended to other users. Also this page allows the management of services related to a user.

4 Implementation

4.1 Database Implementation

OpenSim simulator, mentioned above, includes a database which is essential for simulator's proper working, and in which users, objects, regions and scripts' registration take place in the virtual world. The database has a table named users, in which user IDs (UUIDs) are defined, those IDs are used to link original database tables with new tables needed by the recommender agent in order to meet their goals.

Figure 3 shows how the tables are linked to the existent users table, in one hand it's linked to the user profile table, which in turn is connected to the topics that the user prefers. In the same way, the activities are related to a user by his ID. Finally we store the recommended services' list for subsequent rating.

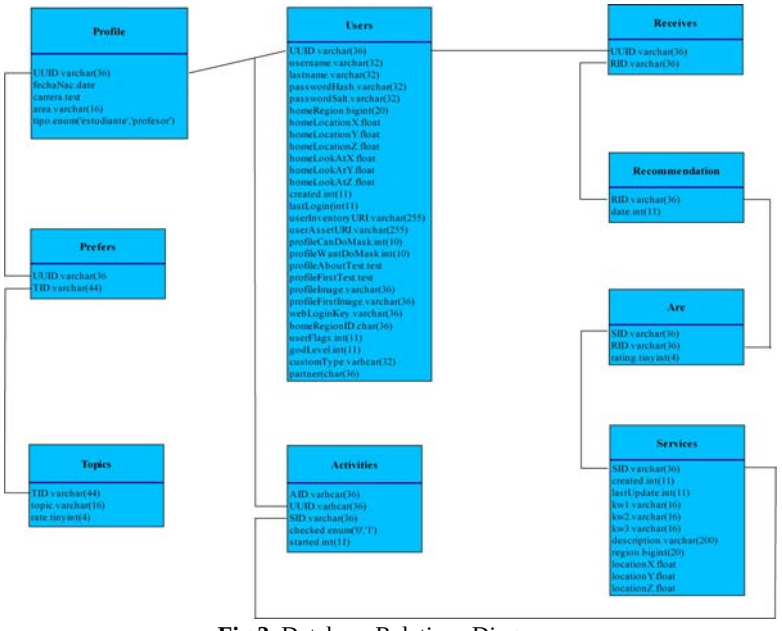

Fig 3. Database Relations Diagram.

Table *Profile* stores the user information, it has a foreign key UUID (users), that is because profile is just an extension of the user information, but the table users is already defined in the original Opensim database and we didn't want to modify those tables.

Table *Topics* store the keywords associated to services, these keywords are the themes that user prefers. This table has its primary key TID, the name of the topic and its rate.

Table *Prefers* connect the table profile and the table topics, it has a primary key which references to UUID (profile) and TID (topics). This way we can now what topics a user prefers.

Table *Services* stores the services created by users and registered in the services web page, its primary key is SID, and has more information as created, description, region, in order to know where the object is.

Table *Activities* registers any interaction between the user and an object (which can be linked to a service), it has a primary key AID, and two foreign keys, UUID (users) and SID (services). These activities are not stored permanently in the database, once the activity has been analyzed, it is marked as "checked" and then deleted.

Table *Recommendation* stores the recommended services list sent to the users, only has its RID and the date the list was sent.

Table *Are* is connected to table Recommendations and Services, it connects the services with a recommendation list. Its primary key references to SID (services) and RID (recommendation).

The last table is Receives, which connects a recommendation list to a user Its primary key references to UUID (users) and RID (recommendation).

4.2 Building user profile

Sensors are objects located along the virtual world, by which they listen the communication channel used in OpenSim (registers interaction between avatars and objects associated to services in simulator). These sensors are programmed using Linden Script Language and the information retrieved by them is sent to a web page in order to store that data in the database, this is because the language is very simple and it doesn't allow connection to databases or output of information into a file. Sending the data to a web page is very helpful, and using PHP (which is a robust language) we can connect to databases.

Sensors register information in table activities in the database, ensuring availability for future analysis. This table is continuously handled by the preferences classifier agent to debug information stored in it. The agent reads this table, and according to the information read, inserts or increases the rating in table topics. After doing this, the agent delete the information previously read in order to avoid garbage data stored in the database.

In figure 4, we show the interaction of sensors, the preferences classifier agent and the tables in the database. This is part of the architecture, but with more details showing the process to build the user profile.

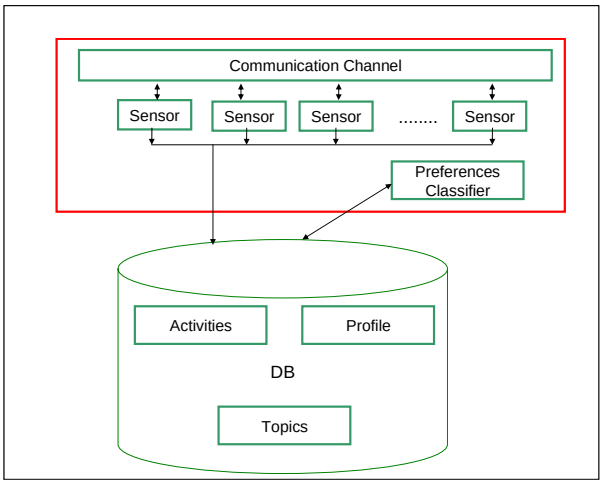

Fig 4. Building a User Profile Architecture

Finally it is worth noting that the sensors inside the virtual world are in a passive form, they only changes to an active form when an avatar is close to them, and then, they starts to collect the information.

4.3 Algorithm for rating recommended services

Tasks related to the preferences classifier agent are divided in two phases: Phase 1 Activities analysis; Phase 2 Rating analysis [5].

In phase 1, the agent continuously monitors database, where services, which avatar has interacted with, are stored, it classifies the services by preference (p) and by length (l). Preference is measured by services accessed frequently by the avatar, and length is measured by the elapsed time since the avatar started to use the service until it finishes using it. The agent applies Algorithm 2 in this phase and Algorithm 3 on phase 2 [5].

We propose new values for the ratings of recommended services in the Algorithm 3. With previous values the rating just increase its value showing only positive preferences, nevertheless using that values we can't know if the user dislike that service or it just doesn't matter to him. Moreover a problem arises just by increasing the rating: "is there a bound for the rating value or could it increase without a limit?"

The previous values were: increase 0.1 if the user rating is 1, increase 0.2 if the user rating is 2, same for rating on 3, 4 and 5, if user rates 0 do nothing. Below we show the algorithm with the new values:

Algorithm 3
Step 1: The agent retrieves services rated by the user in table recommendations
Step 2: Select keyword from table services in order to update or insert it in table topics
Step 3: If rating associated to service is:
0: Decreases rating in 0.3

1: Decreases rating in 0.2
2: Decreases rating in 0.1
3: Increases rating in 0.1
4: Increases rating in 0.2
5: Increases rating in 0.3

The values description is explained in the table 1.

Table 1. Description of the values

Value	Description
0	The service dislikes very much
1	The service dislikes
2	The services dislikes a little
3	The services likes a little
4	The service likes
5	The service likes very much

We can note that, according to the rules for rating services, the rating number can be a negative value. This way we can now know that a user doesn't like certain topic.

4.4 Learning rules

A first approach to learning rules has been developed. The rules try to solve the posterity problem; this is for topics that the user rates as a like a time ago, but now that topic has no more interest for the user. For example: a year ago a user interacted with topics about Olympic games, and that topic has a rate of 1.5, now the user is no more interested in Olympic games, but the recommender agent continues recommending services related to that topic because of its rating. There is not enough (and also not advisable) that the user rate several times the service recommended with 0, in order to decrease its value. We can apply some rules that, given a period of time, analyses if the user has been interested in a topic. If the topic has not been modified in that period of time we can "reset" the value of the topic setting it on a value of 0.

```
If a topic rating has not been modified in X period of time
Then     if its value is negative
         Then do nothing
         Else reset its value to 0.
Else     //a topic rating has been modified in X period of time
         if it has been modified only decreasing it
         Then reset its value to 0.
         Else do nothing.
```

These rules are very simple but solve the posterity problem explained above. More rules will be developed for working analyzing user activities and updating the user profile

5 Results

Web page access to management of services is working properly and there has not had problems. The page is based on Apache server using PHP and MySQL. Services related to users has been added and stored in the data base, and the modification or deletion of them works fine too. Relations of the tables in the database fulfill its role, and no problems have been reported related to it.

Sensors register information read on the communication channel and we are doing some tests to ensure that the web page using PHP stores the data efficiently in the database, one disadvantage of using web page is that the server may be saturated with connections, but at the moment this is the best solution.

The Learning Techniques module is under construction and the learning rules have not been implemented yet, but we hope shortly have some results to show.

6 Conclusions and Future Work

This paper shows an important advance in the implementation of the recommender agent, step by step the different modules are been developed, and they works properly. There has had some complications developing the sensors but they has been successfully solved bypassing the language and the database through a PHP webpage.

Currently the user profile is being developed and analyzing it in order to find the optimal balance between the user preferences and user activities. Agent learning capacities are based on deductive rules, which are being created. Only the rules presented in this work have been developed, but when all the rules are ready, the agent could infer the best services to recommend to the user.

References

1. Marcus, A. “Caracterización de los Servicios de la Educación a Distancia desde la óptica de una plataforma distribuida orientada a objetos”. Master degree thesis ITESM. Monterrey, México., 2000.
2. Daniel Livingstone, Judith Kemp, and Paul Andrews. Sloodle Learning System for Virtual Environments. http://www.sloodle.org, 2009. Accessed on February 23, 2009.
3. Oswaldo Velez-Langs, Carlos Santos. “Sistemas Recomendadores: Un enfoque desde los algoritmos genéticos”,. Industrial Data, ISSN: 15609146, Volumen 9, Fascículo 1, Pag. 20 - 27.

4. Ana Gil, Zahia Guessoum, Francisco García. “Recomendadores en un Sistema Multiagente Adaptativo para el Comercio Electrónico”,. Taller en Sistemas Hipermedia Colaborativos y Adaptativos dentro de las JISBD’2002. El Escorial, 18 al 22 de Noviembre del 2002.
5. Ramirez, J.C., Vilariño, D., López F., (2009). A Recommender Agent Design For Virtual Worlds. Proceedings of the IASK International Conference – E-Activity and Leading Technologies. pp 142-146. ISBN: 978-989-95806-7-1.
6. OpenSimulator Web page. http://opensimulator.org