Multi-agent system model for tutor recommendation in ubiquitous learning environments

Beatriz Fernandez Reuter[1,2], Margarita M. Alvarez[1], Gabriela Gonzalez[1,2], Elena B. Durán[1]

[1]Instituto de Investigaciones en Informática y Sistemas de Información – Facultad de Ciencias Exactas y Tecnologías – Universidad Nacional de Santiago del Estero (UNSE)
Av. Belgrano (S) 1912 – Santiago del Estero – Argentina

[2]UNSE, CONICET, Facultad de Ciencias Exactas y Tecnologías, Santiago del Estero, Argentina.

{bfreuter,alvarez,ggonzalez,eduran}@unse.edu.ar

Abstract. *Ubiquitous learning environments allow to learn anywhere at anytime, enabling people to have better learning experiences in their daily lives. In order to detect learning problems and offer help, a tutor needs to observe the actions of students and to evaluate them, which is not easy to accomplish in a ubiquitous environment. Therefore, this work presents a multi-agent model to generate recommendations of tutors in the topic that a student needs help with, based on the experiences of these tutors with other students, their availability and their physical proximity. The proposed model allows to monitor the student interaction within the learning environment, detect learning problems and offer personalized help.*

1. Introduction

Ubiquitous learning environments overcome the limitations of a traditional classroom or learning environment, in such a way that the idea of learning anywhere at anytime becomes possible. The use of devices such as mobile phones and tablets creates new opportunities for students, as they are constantly connected to mobile networks. This way, educational content can be accessed anywhere and anytime, and learning interactions can take place when and where students need it, in different situations of their daily lives, with no restrictions of either space or time [Graf and Kinshuk 2008].

A ubiquitous learning system should not only provide students with educational resources anywhere and anytime, but it should also offer appropriate help to aid students fulfill their learning activities. In that sense, there are context-aware techniques that consider contextual data such as location, time, and other elements, to generate more successful recommendations. Most of the reviewed works that deal with ubiquitous learning environments focus on the recommendation of educational content, learning paths or peers to conform groups for collaborative learning. However, tutor (teacher, expert in a topic, or advanced student) advice is also important in some learning situations, where a student needs concrete help on some topic.

The integration of a multi-agent architecture in a ubiquitous environment can aid in the detection of learning problems in real time and in the provision of assistance, based on the student's location and the location of those who can help. Besides, in order to achieve a closer flexibility to that provided by a human tutor on-site, the architecture needs to define configurable system settings that allow to fine-tune the criteria used within

DOI: 10.5753/wave.2018.10

the monitoring, help and communication tasks. A multi-agent system is a consortium of multiple agents working together, where each agent is specialized in a particular function [Sajja and Akerkar 2012]. These agents are capable of doing specific tasks in behalf of the users [Nwana 1996]. To do so, they are placed in a concrete environment and execute autonomous actions, in a flexible way, in order to reach their goals [Wooldridge 2002]. This flexibility can be translated as [Wooldridge and Jennings 1995]: reactivity or capacity to perceive its environment and act promptly to changes in that environment; pro-activity or capacity to exhibit a goal-oriented behavior, and to take the lead; and social ability or capacity to interact with other agents (possibly humans) through a communication language. In educational applications, these agent properties can be reflected in collaborative tasks, such as providing an intelligent interactive medium between members of a team, in the personalization of content and help, by constantly observing actions of students and tutors, and in the integration between students and teachers, allowing a better communication between them [Bokhari and Ahmad 2014].

In [Duran and Alvarez 2017], a method to generate recommendations of experts in the subject that a student wants to learn has been proposed. This method takes into account the experiences of these experts with other students, their availability and their physical closeness with the student for face-to-face consultations. In this work, a model of a tutor recommendation system is proposed. This model is based on a multi-agent architecture to monitor the student interaction with the system, detect learning problems and offer personalized help. To generate the recommendations, the method proposed at [Duran and Alvarez 2017] is applied.

The paper is organized as follows. The following section reviews works related to this proposal, while section 3 describes the multi-agent model proposed. The last section presents the main conclusions and future work.

2. Related Work

Several proposals of multi-agent systems for ubiquitous learning environments have been found in the literature. A description of the most relevant ones is provided below.

A multi-agent system that provides support for learners using mobile devices in a hospital u-learning environment is proposed in [Boudabous et al. 2017]. The focus of this work is the provision of personalized educational contents on demand. To do so, the system gathers information from the learner, which can be a patient or a nurse, such as his profile, location and blood pressure, and delivers personalized educational resources according to his data. The teacher, a doctor, is in charge of creating the different types of learning resources that can be accessed by the students. The system architecture is comprised of three layers, each one with a different set of agents. The user layer has learner, teacher and tutor agents, to communicate with the user. The core layer is where the main processing occurs, and hosts an admin agent, a profile manager agent and a content manager agent. Finally, the cloud layer handles the storage and execution, with three types of agents: store agent, deliver agent and find agent. The architecture also includes free agents (link agents) to communicate each layer with others.

In [Salazar et al. 2015] a multi-agent context-aware u-learning system that offers adaptive virtual course planning, personalized course evaluation, selection of learning objects according to student profile, search of learning objects in repository federations,

search of thematic learning assistants, and access of current context-aware collaborative learning activities involved, is presented. The system architecture is composed of six types of agents: user, recommender, searcher, planner, evaluator and awareness agent, and it is programmed in JADE, a Java Agent Development Framework. The thematic learning assistants recommendation offers students the possibility of searching advanced students that are available, are spatially close and have knowledge of the student's areas of interest.

[Ayoola and Mangina 2014] present PULP, a personalized ubiquitous learning platform. The learning environment is designed to provide personalized content, anytime and anywhere access and collaborative activities and services. The system is programmed in JADE and has four types of agents: mediator agent, performance agent, interest agent and recommendation agent. The personalization is based on a users' profile, which contains data about his preferences and academic strength. There are no specific details regarding the collaborative activities and services provided.

In [de M. Neves et al. 2014] an agent-based architecture for context-aware and personalized event recommendation in the pervasive university domain is presented. There are three types of agents: access control, recommender and location agent. The recommender agent takes into account the user profile and other contextual data, such us user location and date, to choose between all available cultural and academic events on campus, and to provide a filtered list of personalized events.

[Garcia-Cabot et al. 2014] present a multi-agent system that focuses on adaptation of learning content to the kind of mobile device, the context and the competences of each learner. The system architecture includes five types of agents that collaborate to reach the personalization goal. First, a logical sequencing agent establishes a sequence of the topics or subjects that the learner has in his/her syllabus. Then, a federated search agent compiles a list of potential learning objects to be used, by searching in different repositories using keywords according to the topics of the learner syllabus. Finally, a device agent filters the learning objects that the learner's mobile device does not support, and a context agent sorts them by context. These four types of agents are administered by a manager agent, which also interacts with the learner to gather and present information.

In [Tan et al. 2009] a multi-agent system architecture with several adaptation goals is presented. On one hand, the system allows students to create location-based learning content and share it with other peers. It also creates learning groups dynamically by identifying the individual mobile learners' geographic locations and other learning factors, such as learning history, previous learning performance, educational level, learning style and learning interests. Additionally, all content presented to the learner is adapted according to the characteristics of his mobile phone. In order to achieve this goals, there are three types of agents that cooperate in the collaborative platform. The learner agent is installed into students' cellular phones and is responsible for reporting students' location information to the location-aware agent, as well as cooperating with the resource agent to manage the authentic examples. The grouping algorithm is managed within the location-aware agent.

3. Multi-agent system architecture

The multi-agent system for tutor recommendation is designed to be integrated in a u-learning environment based on learning objects (LO). This allows to monitor a student in order to detect problems in his learning process, to recommend tutors, either physically close to him or available on-line, and ultimately, to provide a communication channel with them. To realize the tutor recommendation, the system takes into account context data and the tutors' amount of attended consultations. A tutor can be:

- *A teacher:* who offers tutoring services for all topics from his courses.
- *An expert:* who offers tutoring services for some topics of his specialty.
- *A student:* who offers tutoring services for all the topics of the courses he is attending, as long as he satisfies the student tutoring conditions.

The agents carry out their activities based on a set of configurable system settings that are defined by each teacher according to his own assessment and evaluation criteria. As can

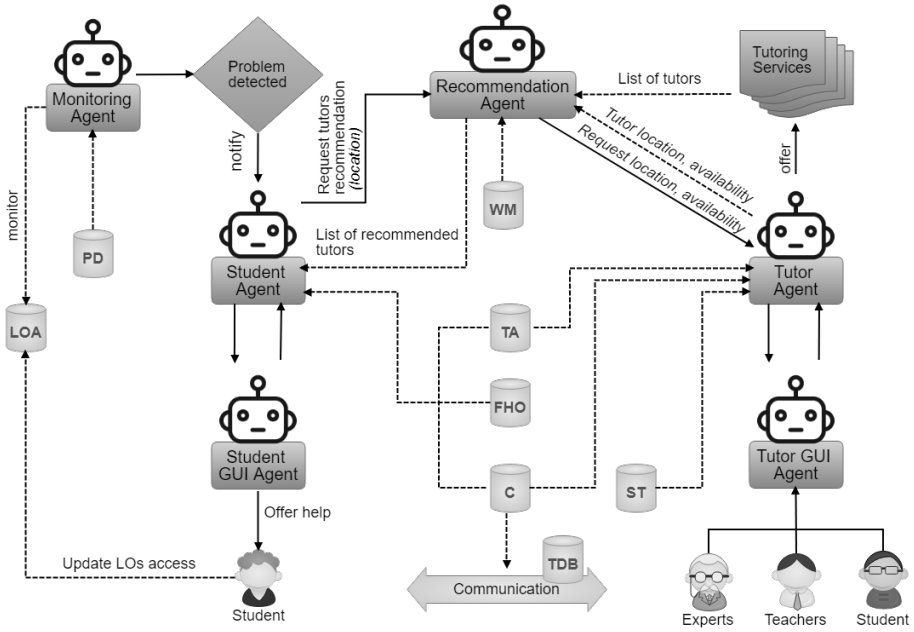

Figure 1. Multi-agent system architecture

be seen in figure 1 the system has the following components:
Monitor Agent: this agent is in charge of monitoring the student's access to the course LOs in order to detect problems in a topic, according to the problem detection settings. Once a problem has been detected, it notifies the *student agent* of that student.
Student Agent: this agents determines the student location, based on his cellphone GPS data and manages the corresponding help. To do so, it notifies the *recommendation agent* the topic (t) that the student needs help with, and the location (sl) of that student. In addition, once the recommended tutors list is received, it is in charge of setting up communication channels with each tutor.
Tutor Agent: this agent starts when a tutor logs in the system and automatically publishes his tutoring services in the corresponding topics. If the tutor is a student, it will publish services as long as the student tutoring conditions are met. This agent is also responsible

Table 1. Tutors Weight Matrix

Student / Tutors	T_1	T_2	...	T_m
S_1	w_{11}	w_{12}		w_{1m}
S_2	w_{21}	w_{22}		w_{2m}
...				
S_n	w_{n1}	w_{n2}		w_{nm}

of detecting the tutor's location, based on their GPS cellphone data as well as the availability for consultation in that moment. It also provides these data to the *recommendation agent* when requested, and it sets up the communication with the students that require it, according to the tutors consultation and communication settings.
Tutoring Services: this is a list of tutor agents that offer tutoring services. It contains the following information: for each tutor that offers services, his name and the list of topics that he can be consulted about.
Transaction Data Base (TDB): a database containing the information about the consultations made by students and the assistance provided by tutors.
Weight Matrix (WM): this matrix is generated daily based on data from the *transaction DB*, in the following way:
the preferences of each student *(i)* regarding a tutor are expressed as a weighted vector $< w_{i1}, w_{i2}, ..., w_{in} >$ where each w_{ij} is calculated according to formula 1.

$$w_{ij} = \frac{x}{y} \tag{1}$$

where x equals the number of times that tutor j advised student i and y equals the total number of times that student i was advised on this topic by any tutor. This information is obtained from the *transaction DB*. If the weighted vectors of all students making up set S are considered, it is possible to construct the *weight matrix* (Table 1) using the preferences of the learners that already worked on the topic related to the tutors on this topic.
Recommendation agent: this agent receives assistance requests from the student agents. Based on the published tutoring services, it selects tutors in the topic (t) and builds the set T. Then for each tutor of the set T it requests the current location and the availability from the tutor agent. The recommendation agent is responsible of generating two kinds of recommendations: one for consultations with tutors who are physically close and another one for consultations with on-line tutors.

Physically close tutors recommendation

Distance Vector Generation: In order to consider the contextual characteristic (location), the distance (d) in meters between each tutor T_j of the set T, and the learner's current location is calculated. That way, the vector of distances $< d_1, d_2, ..., d_m >$ is obtained. From this vector, the weights $\frac{1}{d}$ are obtained. They allow to value the tutors closer to the learner with higher weight, and those that are farther with less weight.

Calculation of scoring function: this function allows to obtain a vector $< TP_1, TP_2, ..., TP_n >$ using the weighted values for each tutor on the topic, obtained from

WM. It is calculated with equation 2.

$$TP_j = \left(\sum_{i=1}^{m} w_{ij} \right) \frac{1}{d_j}; j = 1...n \tag{2}$$

Then, the vector is ordered from high to low, with which the recommendation of tutors to the learner is generated.

On-line tutors recommendation

To generate on-line tutors recommendations, the scoring function is applied excluding the distance-based weighting, and therefore calculated as specified in equation 3.

$$TP_j = \left(\sum_{i=1}^{m} w_{ij} \right); j = 1...n \tag{3}$$

In both cases the vector TP is sent to the student agent.

Learning Object Access (LOA): this database registers the access of each student for each LO content as well as the final status of the LO (pass/not pass).

Interface Agents: these agents are responsible of showing the graphic user interfaces to the users and they are called by student and tutor agents, which provide the information to be displayed.

System settings: these are the configurable conditions of the model. They are specified by each teacher for his courses and represent the conditions of the monitoring activities, help offering and communication policies.

- ***Problem Detection (PD):*** defines the conditions to detect that a student has learning issues in a topic:
  - *LO permanence time* must be higher that the maximum learning time indicated in the LO meta-data.
  - *LO number of accesses* must be higher that the maximum number of times stipulated by the teacher.
  - *Grade obtained* by the student must be equal to the value assigned by the teacher.
- ***Student tutoring (ST):*** defines the conditions that must be fulfilled by the student to offer tutoring services in a course topic.
  - The student assessment in the LO is equal or higher than the value established by the teacher.
- ***Tutors Assignment (TA):*** specifies how to distribute the help between the available tutors.
  - The maximum number of consultations that a tutor can attend is defined by the teacher.
- ***Frequency of Help Offering (FHO):*** specifies when to offer help to the student again, if there's a problem detected in the same topic.
  - Help will be offered if the student accesses the LO a number of times greater than those specified by the teacher, given that he has already postponed or canceled previous help.
  - The student has the possibility of disabling the system recommendation function, in which case help will not be offered until he activates it again.

- ***Communication (C):*** defines guidelines to establish communication between students and tutors. It can be synchronous through chat, o asynchronous through the use of e-mail or forums.
  - The communication is established according to the modality, duration and characteristics established by the teacher.

4. Conclusions and Future Work

New learning models foster autonomous, contextualized, and personalized learning. Thus, we consider the support that can be provided to students through technology, to be very relevant because one of the key characteristics of an ubiquitous learning system is to offer appropriate help to aid students fulfill their learning activities. Most of the reviewed works that deal with ubiquitous learning environments focus on the recommendation of educational content, learning paths or peers to conform groups for collaborative learning. However, teacher, expert, or advanced student advice is also important in some learning situations, where a student needs specific help on a particular topic. Therefore, in this work we integrate a multi-agent model to a u-learning environment as a way to provide an efficient medium to detect learning problems of students and to recommend personalized help, due to the autonomous, reactive and proactive capability exhibited by agents.

Besides, the implementation of a tutor recommendation method enriches the model, allowing to combine the physical distance between the tutor and the student with the popularity achieved by each tutor, to provide a personalized and contextualized recommendation strategy.

Likewise, the inclusion of variable system settings allows to configure the model according to the particular needs of each learning environment and teacher, providing greater flexibility in the monitoring and recommendation tasks.

Currently, we are developing a prototype to implement the model and test it in a real-world learning environment.

References

Ayoola, O. L. and Mangina, E. (2014). Personalisation of a u-learning environment for third level education. *Yükseköğretim Dergisi*, 4(1):54–60.

Bokhari, M. U. and Ahmad, S. (2014). Multi-Agent Based E-Learning Systems : A Comparative Study. In *International Conference on Information and Communication Technology for Competitive Strategies . ICTCS '14*, pages 1–6.

Boudabous, S., Kazar, O., and Laouar, M. R. (2017). Towards a hospital ubiquitous learning system based agents. In *Proceedings of the International Conference on Computing for Engineering and Sciences*, ICCES '17, pages 56–62, New York, NY, USA. ACM.

de M. Neves, A. R., Álvaro Marcos G. Carvalho, and Ralha, C. G. (2014). Agent-based architecture for context-aware and personalized event recommendation. *Expert Systems with Applications*, 41(2):563 – 573.

Duran, E. B. and Alvarez, M. (2017). Method for Generating Expert Recommendations to advise students on ubiquitous learning experiences. In *Conferencia Internacional de la Sociedad Chilena de Ciencia de la Computación (SCCC 2017)*.

Garcia-Cabot, A., Garcia-Lopez, E., De-Marcos, L., Fernandez, L., and Gutierrez-Martinez, J.-M. (2014). Adapting learning content to user competences, context and mobile device using a multi-agent system: case studies. *International Journal of Engineering Education*, 30(4):937–949.

Graf, S. and Kinshuk (2008). Adaptivity and Personalization in Ubiquitous Learning Systems. In *4th Symposium of the Workgroup Human-Computer Interaction and Usability Engineering - USAB 2008*, pages 331–338, Graz, Austria. Springer 2008.

Nwana, H. (1996). Software Agents: An Overview. *Knowledge Engineering Review*, 11(3):205–244.

Sajja, P. S. and Akerkar, R. (2012). *Intelligent Technologies for Web Applications*. Chapman & Hall/CRC, 1st edition.

Salazar, O. M., Ovalle, D. A., and Duque, N. D. (2015). Adaptive and personalized educational ubiquitous multi-agent system using context-awareness services and mobile devices. In Zaphiris, P. and Ioannou, A., editors, *Learning and Collaboration Technologies*, pages 301–312, Cham. Springer International Publishing.

Tan, Q., Kinshuk, Kuo, Y. H., Jeng, Y. L., Wu, P. H., Huang, Y. M., Liu, T. C., and Chang, M. (2009). Location-based adaptive mobile learning research framework and topics. In *2009 International Conference on Computational Science and Engineering*, volume 1, pages 140–147.

Wooldridge, M. (2002). *Introduction to MultiAgent Systems*. Wiley & Sons, second edi edition.