ADCAIJ: Advances in Distributed Computing and Artificial Intelligence Journal
Regular Issue, Vol. 8 N. 1 (2019), 5-12
eISSN: 2255-2863
DOI: http://dx.doi.org/10.14201/ADCAIJ201981512

Learning process: Multi-Agent Tutoring System

Manuel Pérez Moríñigo, Víctor Merchán Montero, and José Luis Martín Pérez

manuperez@usal.es, vmerchan@usal.es, pepe9817@usal.es
University of Salamanca

KEYWORD

Multiagent systems; education; tutorship; scheduling

ABSTRACT

A multi-agent architecture has been developed for tutorial assignation scheduling. It has two main types of agents: the students and the teachers. These two are coordinated by an algorithm which assigns the classes in order of arrival. The architecture will provide the necessary tools to the students, so they get the maximum profit from the tutorials. Students and Lecturers can coordinate their tutorial meeting in an efficient way with the help of the multi-agent system.

1. Introduction

Knowledge has become a very important element in our society and has taken a vital paper in the progress of our society. Nowadays, the amount of information available for everyone with an internet access is uncountable and it is essential to know how to manage it. This, combined with the spread of Networks such as LANs (Local Area Network) and WANs (Wide Area Networks) have made a very easy and rich access to information in a world-wide level.

Because of this improvement of the technologies for the spread of all this information, the learning process bounded must make the same advance (BECERRA-BONACHE *et al*., 2014). There are certain platforms such as e-Learning or distance learning academies which try to give some help to the students which can't afford face-to-face classes or just prefer the facilities that distance learning give them. Among all the possibilities and platforms that are available for this service, there are still plenty of improvements that can be implemented to upgrade this system (Rodrigues *et al*., 2013).

Our model proposes something similar with some differences which provide the student more facilities such as a more autonomic way of learning for the him/her by giving them the independence of a wide-open schedule. In this platform, the student asks for the tutorships of the subjects that he wants and whenever the teacher is available he chooses the hour that best fits with his/her preferences (SILVEIRA *et al*., 2016). This system uses a multi-agent system formed by two main types of agents: students and teachers that have different functions. According to Russell and Norvig (1995) "An agent is anything that can be viewed as perceiving its environment through sensors and acting upon that environment through effectors".

The proposed multi-agent system is composed of several of these agents that work together and coordinate their actions for toturial scheduling.

The MAS runs an algorithm that assigns the tutorship for one or another student attending to some criteria such as the preferences of him, who asked before for the tutorship at that time and the availability of the teacher of that subject. The system will also

facilitate the long-distance studies by supplying tutorships with the teachers of the subjects taken by them.

Even though, the main objective of this is trying to develop a system which improves an ambit of the educational system as the tutorships, as the actual system is not well organised, and the students don't get the maximum profit.

The state of the art in this field is presented and the proposed architecture outlined. Results and conclusions are finally presented.

2. State of the art

Multiagent systems are systems composed of multiple interacting computing elements, known as agents. Agents are computer systems with two important capabilities.

In this kind of systems there are many kinds of agents and each one delivers a different task. Each task solves an individual problem. The set of all of them would have been impossible to solve for just one agent.

In recent years, multiagent systems have been used in many different fields. They have managed to introduce themselves in the business very quickly, thanks to their great versality.

We can choose video games as an example to prove this. More specifically in Sandbox games, where multiagenting is quite used to make environments come alive. *Middle-earth: Shadow of Mordor* is a good example of what a sandbox war game can offer. *Monolith* has developed a world where we can interact almost with anything. For example, enemies remember when they kill you, and if they do, they level up and get stronger.

As we see, there are plenty kinds of agents. In our project, we are particularly interested in intelligent agents. Many authros have been working on this area and here it is presented a summary of the most interesting proposal in this field.

The project proposed by Webber *et al.* (2000) develops a theoretical and methodological foundations to guide the computer modelling and the concept of learning environments. The Baghera platform is based on the principle that educational function is a property of the interactions between its components.

Capuano *et al.* (2000) presents a multiagent system architecture based on distance learning. The teachers upload their lessons and the students can see the online content. Despite of many people don't have a place to study, this system tries to solve the problem of learning. In Capuano *et al.* (2001) it is introduced an innovative architecture for Intelligent Tutoring making use of Software Agents. The way of representing and storing knowledge is discussed together with the ability of the system to manage individual learning paths for different users.

These authors have investigated other alternatives and evolve their initial model. For them, learner is composed by a cognitive state that measures the knowledge reached by him at a given time and by a set of learning preferences that provide an evaluation of which learning strategies are more feasible for him (Capuano *et al.*, 2012).

Gascueña *et al*., (2005) present an application of an agent-based for Intelligent Tutoring Systems formed by three main components (the Student Mode, Domain Model and Pedagogical Model). Also, they added an Educational Model which provides mechanisms to the teacher needs.

Greer *et al*. (2001) explains a study from several large-scale world deployments of the I-Help agent-based learning system. The authors of the paper divide this lessons into two main categories: software engineering lessons and usage lessons.

Zapata-Rivera and Greer (2001) present also a SModel server, a student modelling server used in distributed multi-agent environments. SModel includes a student model database and a Bayesian student modelling component and provides several services to a group of agents in a CORBA platform.

Turgay (2005) has developed a distance learning environment with flexible and cooperation features. The system agents are teachers and students and it also has a component resources.

Webber and Pesty (2002) show a multi-agent system for a distance learning environment based on a who-level agent architecture. The higher level of MAS (multiagent systems) is composed by cognitive agents, while the lower level is formed by reactive agents Finally Sencer (2008) presents and interesting study on an agent based and databased learning system which supports the decision mechanism within the system.

3. Proposed architecture

Now, we are going to present the architecture structure. The proposal model consists of two types of classes: responses and requests. The requests have as attributes: the month in which we are, the number of student who sent it, the day and time of the tutorship and subject to be dealt with in that tutorship. The responses have as attribute: the available hours of teacher, the request the student has ordered to the teacher and the state of the request. The state indicates if the request has been accepted, rejected or if it's in transit.

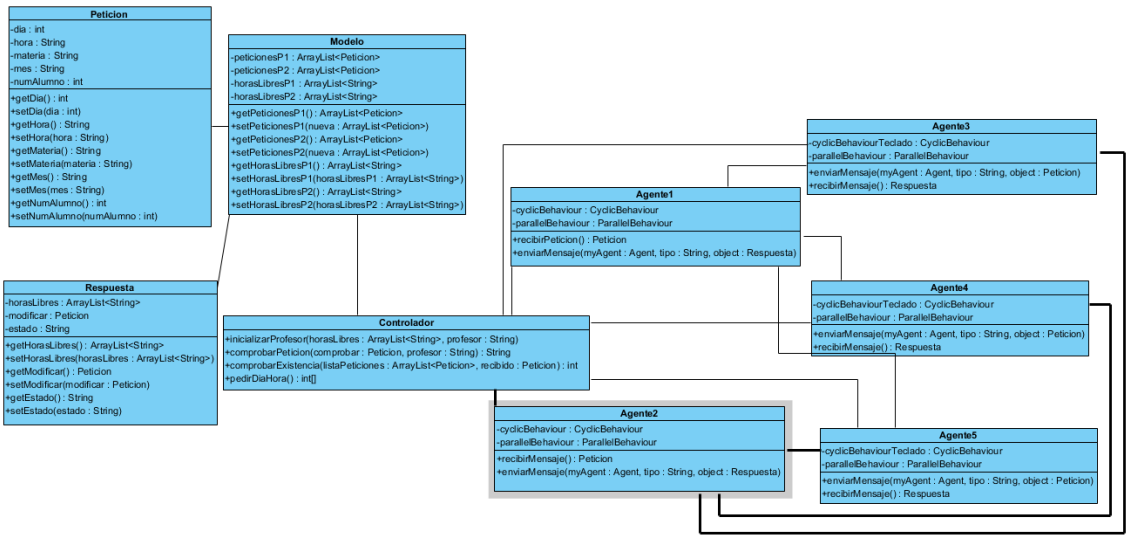

Figure 1: Class Diagram

As student agent 's requests arrive, the agent teacher will accept or reject it. When the request is accepted, it will be added to the teacher's request list. In the case that the agent has asked for an hour and the teacher hasn't any tutorship for that hour, he will send his timetable to the student agent. The student agent will have to choose an hour of the timetable, which is stored in the model.

The student will ask for a tutorship. The agent student in charge of this student will ask him for the tutorship data. The student will send the data to agent student, who will process the data. If the data are correct, it will send the request to teacher agent in charge of the subject that the student has asked for.

If the hour is in the schedule, the agent teacher will send to the agent student if the request has been accepted or not. If the request is in the list of requests the agent teacher will send the answer in the status of "Denied" and if it's not, will send the answer with the status of "Accepted". Finally, the student will have the assigned tutorship.

If the student chooses an hour that it's not in the teacher's timetable, the agent teacher will send the agent student the schedule of himself. Afterwards, the agent student will show that timetable to the student and he will chose the hour that best fits to him and will send this data to the agent student, who will send it to the teacher. If the request is in the list of requests the agent teacher will send the answer in the status of "Denied" and if it's not, will send the answer with the status of "Accepted". Finally, the student will have the assigned tutorship.

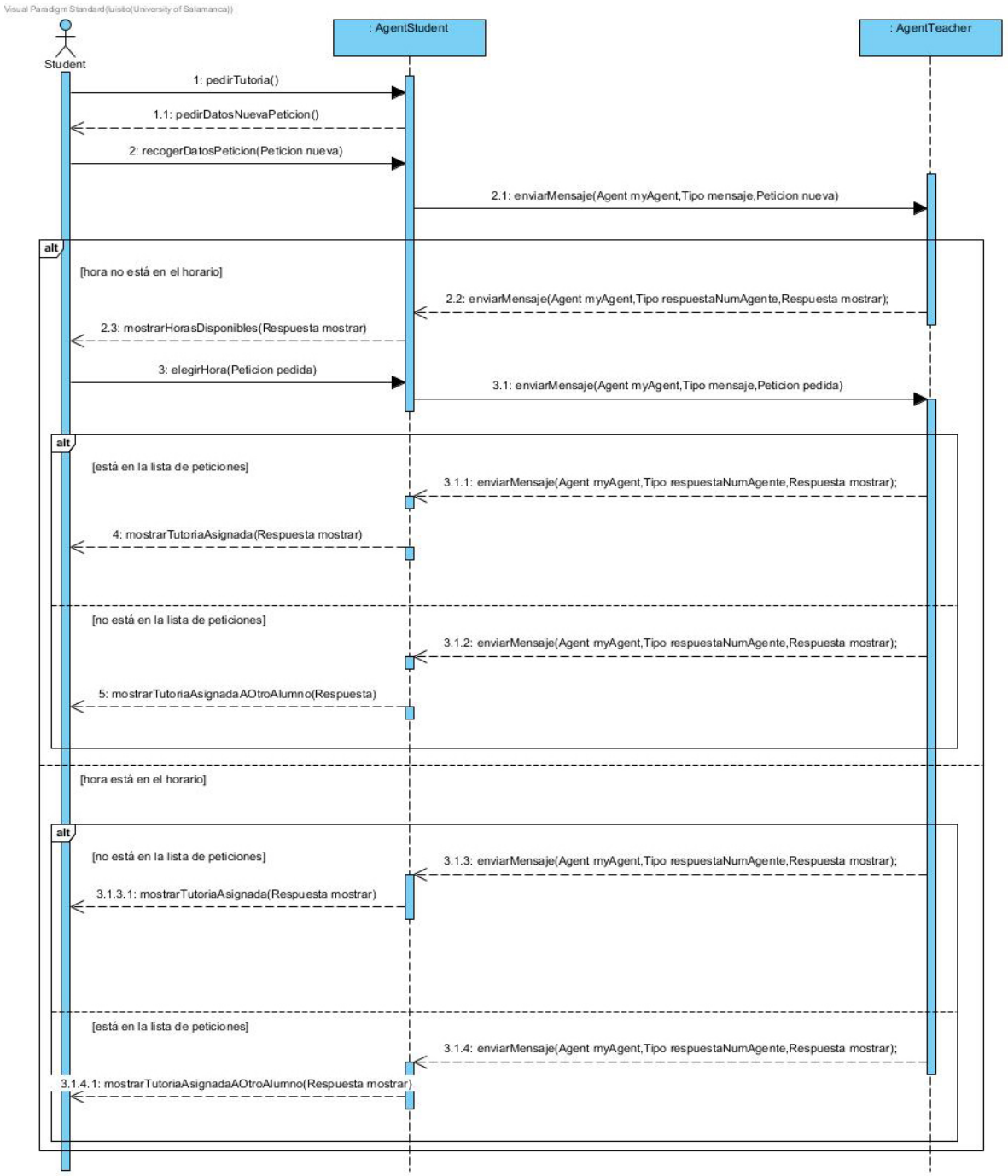

Figure 2: Sequence Diagram

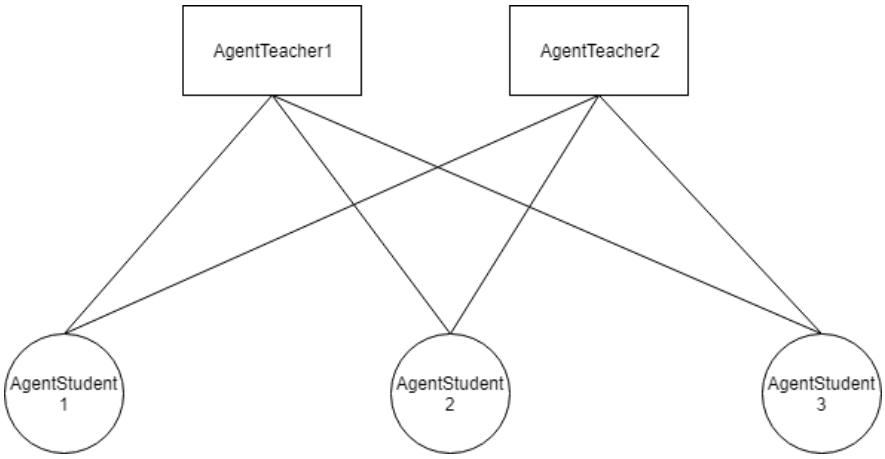

Figure 3: Diagram of agents

In case that the agent teacher is not available, the system will send a message with the status of "In process" to the agent student with the timetable of the teacher. The agent student will ask the student to choose a new hour inside the timetable of the teacher and when he/she has chosen it, the request will be sent to the agent teacher with the new hour for the tutorship. Afterwards, when the agent teacher receives it, will check that the new hour doesn't belongs to another student in that day of the month. If so, the request will be added to the requests of the teacher and it will be sent a message to the student with the status of "Accepted" and the request of the same student with the data of it (month, hour, teacher and subject). In the other hand, in case that the new hour requested was conceded before to another student, the message sent to the student will contend the status of "Denied" and another request to the student agent to show the student that the hour selected by him/her was chosen before by another student.

Otherwhise the agent teacher will check that the new hour doesn't belongs to another student in that day of the month. If so, the request will be added to the requests of the teacher and it will be sent a message to the student with the status of "Accepted" and the request of the same student with the data of it (month, hour, teacher and subject). In the other hand, in case that the new hour requested was conceded before to another student, the message sent to the student will contend the status of "Denied" and another request to the student agent to show the student that the hour selected by him/her was chosen before by another student.

4. Experimental results

In this experiment it has been used a total of 5 agents where 2 of them are online teachers and the rest are students. These students have the possibility of asking for a tutorship whenever they need it. In case that two or more students ask for the same hour for the same subject, the system will assign that tutorship to the first one who asked for it and will send a message to the other one with the resting free hours for that subject. This algorithm is in charge of facilizing the student this access to tutorship by analysing the timetable of all the agents and adapting the hours for all of them.

```
Agent container Container-1@169.254.4.213 is ready.
--------------------------------------------
Tecle el dia que desea pedir la cita, entre los dias 28 y 30:
```

Figure 4: Result output view 1

Here we can see how the agent has initialized his execution and is prepared to execute his actions. The system is asking the student to introduce the day between the ones indicated. If the day introduced is not included in that interval, the system will send an error message and will ask again to introduce the same data. Also, if the user introduces a no numeric character, it will send a message explaining that the character introduced is not a valid one and also will ask to introduce one again the data.

```
Agent container Container-1@169.254.4.213 is ready.
---------------------------------------------
Tecle el dia que desea pedir la cita, entre los dias 28 y 30: 28
Teclee la hora que desea, entre las 0:00 y las 24:00: a
Solo puedes introducir numeros enteros
Teclee la hora que desea, entre las 0:00 y las 24:00: 12.5
Solo puedes introducir numeros enteros
Teclee la hora que desea, entre las 0:00 y las 24:00: 12,4
Solo puedes introducir numeros enteros
Teclee la hora que desea, entre las 0:00 y las 24:00: 10
Introduzca la asignatura para tratar la tutoria: MATEMATICAS
Le ha sido concedida la tutoria el dia 28 a las 10:00 de MATEMATICAS
```

Figure 5: Result output view 2

In this capture, we can appreciate that when the user introduces an invalid character it appears the error message that we described before. If it appears any of the errors explained before, the treatment is exactly the same. Finally, we can also see that if all the data were introduced correctly, we end the execution of the program by sending a confirmation message to the student.

```
Tecle el dia que desea pedir la cita, entre los dias 28 y 30: 28
Teclee la hora que desea, entre las 0:00 y las 24:00: 10
Introduzca la asignatura para tratar la tutoria: Matematicas
La turoria del dia 28 a las 10:00 de Matematicas esta cogida por otro
```

Figure 6: Result output view 3

Here we can appreciate that if the student introduces the data from a tutorship that has been already assigned, the system will send him a message informing about it. We can also see that the name of the subject can be introduced in caps or lowercase, because the system will understand both.

```
Agent container Container-1@169.254.4.213 is ready.
---------------------------------------------
Tecle el dia que desea pedir la cita, entre los dias 28 y 30: 29
Teclee la hora que desea, entre las 9:00 y las 24:00: 8
Teclee la hora que desea, entre las 9:00 y las 24:00: 12:30
Solo puedes introducir numeros enteros
Teclee la hora que desea, entre las 9:00 y las 24:00: 20
Introduzca la asignatura para tratar la tutoria: Biologia
El profesor no tiene ninguna tutoría a esa hora
1) 9:00
2) 11:30
3) 12:45
Elija una de las horas disponibles:2
Le ha sido concedida la tutoria el dia 29 a las 11:30 de Biologia
```

Figure 7: Result output view 4

```
INFORMACIÓN: ----------------------------------------
Agent container Main-Container@169.254.4.213 is ready.
------------------------------------------
nov 28, 2018 12:40:32 AM jade.core.PlatformManagerImpl localAddNode
INFORMACIÓN: Adding node <Container-1> to the platform
nov 28, 2018 12:40:32 AM jade.core.PlatformManagerImpl$1 nodeAdded
INFORMACIÓN: --- Node <Container-1> ALIVE ---
nov 28, 2018 12:40:39 AM jade.core.PlatformManagerImpl localAddNode
INFORMACIÓN: Adding node <Container-2> to the platform
nov 28, 2018 12:40:39 AM jade.core.PlatformManagerImpl$1 nodeAdded
INFORMACIÓN: --- Node <Container-2> ALIVE ---
Tutoría asignada por el profesor 1
Tutoría asignada por el profesor 2
```

Figure 8: Result output view 5

We can appreciate that when the teacher has not available the hour that the student has asked for a certain subject, the system will send a message informing him of that no availability and will propose to choose between the available hours of that same teacher. Afterwards, the user will select the hour that he/she prefers, but not writing the hour but choosing the number of the option selected. The error treatment is such as the previous cases.

We can appreciate that the tutorships have been assigned correctly.

5. Conclusions and Discussion

This article presents a multi-agent architecture based on a class timetable. We have proposed an alternative architecture by Nabeth *et al*., 2005 - "InCA: A Cognitive Multi-Agents Architecture for Designing Intelligent & Adaptive Learning Systems", which is based in a tutorship system, which allows the students ask for tutorships to the teachers in an easy way.

The architecture is a simple one with two types of agents: teacher and student, in which the student agents send requests to the agent teachers and these last decide if they can assign them the tutorship nor not.

The architecture consists of two types of agents: teachers and students. The implementation of the system has a total of 5 agents, two teachers and other 3 which are students.

The architecture has two types of classes: responses and requests. The requests have as attributes: the month in which we are, the number of student who sent it, the day and time of the tutorship and subject to be dealt with in that tutorship. The responses have as attribute: the available hours of teacher, the request the student has ordered to the teacher and the state of the request. The state indicates if the request has been accepted, rejected or if it's in transit.

The vision of the proposal was basically building an architecture where the students and the teachers could put in contact immediately to give the students more opportunities of asking doubts, asking for help in some concrete task that is doing or even sending the practices to the teacher for the future resolution during the tutorship.

Thanks to the multiagent systems, we could build an architecture that attend the necessities of people involved in the educational system. This is only one of the multitude applications that have this kind of systems, the ones which are trying to facilitate complex tasks to our day a day.

This system has also a wide margin of improvements. First, one of the improvements could be that the student could ask for a tutorship with more anticipation, leaving him/her a great margin to organize their schedule (margin of a month or so).

Another one, would be that the teacher could access to the tutorship schedule and delete them if he wants or change them by sending an email to the student.

The experiment is carried out because the article in which it is based treats the intelligent systems applied to the educational ambit. Although the proposal is based on the educational system, it also takes another direction in that sense. The result of our proposal is a tutorship system with a very brighten future having in consideration the time of the creation of it.

This system has also a wide margin of improvements. First, one of the improvements could be that the student could ask for a tutorship with more anticipation, leaving him/her a great margin to organize their schedule (margin of a month or so). Another one, would be that the teacher could access to the tutorship schedule and delete them if he wants or change them by sending an email to the student.

The results are reasonable, but they could get improved adding some new functionalities to the system. Functionalities such as adding a data base to each agent teacher in order to know the schedules and the subjects that they impart in each moment or also adding the functionality that when a student asks for a tutorship hour that has been assigned previously to another student, the system could send him a timetable with the free hours for that subject and that professor in that same day. Although there are also other ambits where the proposal could improve, starting off this system it could me made a system able to put in functioning independent of the level of education that it is applied.

6. References

Capuano, N., De Santo, M., Marsella, M., Molinara, M., and Salerno. S., 2001. A Multi-Agent Architecture for Intelligent Tutoring.

Capuano, N., Mangione, G.R., Pierri, A., and Salerno. S., 2012. Learning Goals Recommendation for Self-Regulated Learning.

Capuano, N., Marsella, M., and Salerno. S., 2000. ABITS: An Agent Based Intelli-gent Tutoring System for Distance Learning.

Gascueña, J. M., and Fernández-Caballero, A., 2005. An Agent-Based In-telligent Tutoring System for Enhancing E-Learning / E-Teaching.

Greer, J., McCalla, G., Vassileva, J., Deters, R., Bull, S., and Kettel, L., 2001. Lessons Learned in Deploying a Multi-Agent Learning Support System: The I-Help Experience.

Sencer, S., 2008. Query Based Learning in Multi-Agent Systems.

Turgay, S., 2005. A multi-agent system approach for distance learning architecture. 1303-6521 volume 4 Issue 4 Article 3.

Webber, C., Bergia, L., Pesty, S., and Balachef, N., 2000. The Baghera project: A multi-agent architecture for human learning. http://julita.usask.ca/mable/webber.pdf

Webber, C., Pesty, S. 2002. A two-level multi-agent architecture for a distance learning environment.

Zapata-Rivera, J.D., and Greer, J., 2001. SMODEL Server: Student Modelling in Distributed Multi-Agent Tutoring Systems. International Conference on Artificial Intelligence in Education AIED 2001. 446-455.

Nabeth, Thierry & Angehrn, Albert & Razmerita, Liana & Roda, Claudia, 2005. InCA: a Cognitive Multi-Agents Architecture for Designing Intelligent & Adaptive Learning Systems. Comput. Sci. Inf. Syst.. 2. 99-114. 10.2298/CSIS0502099N.

Alonso Rincón, Ricardo S.; Prieto Tejedor, Javier; García Pérez, Óscar and Corchado Rodríguez, Juan M., 2013. Collaborative learning via social computing. Frontiers of Information Technology & Electronic Engineering. 2019, Florentino. E-learning Platforms and E-learning Students: Building the Bridge to Success. ADCAIJ: Advances in Distributed Computing and Artificial Intelligence Journal, Salamanca, v. 1, n. 2, p. 21-34, jul.. ISSN 2255-2863.

Becerra-Bonache, Leonor and Jiménez López, M. Dolores, 2014. Linguistic Models at the Crossroads of Agents, Learning and Formal Languages. ADCAIJ: Advances in Distributed Computing and Artificial Intelligence Journal, Salamanca, v. 3, n. 4, p. 67-87, dec.. ISSN 2255-2863.

Ricardo Silveira, Guilherme Klein da Silva Bitencourt, Thiago Ângelo Gelaim, Jerusa Marchi, and Fernando de La Prieta, 2016. Towards a Model of Open and Reliable Cognitive Multiagent Systems: Dealing with Trust and Emotions. ADCAIJ: Advances in Distributed Computing and Artificial Intelligence Journal, Salamanca, v. 4, n. 3, p. 57-86, jun.. ISSN 2255-2863.