An Intelligent Tutoring Systems Integrated with Learning Management Systems

Cecilia E. Giuffra P., Ricardo Azambuja Silveira[1], and Marina Keiko Nakayama[2]

[1] Departamento de Informática e Estatística, Universidade Federal de Santa Catarina (UFSC), Florianópolis, SC, Brasil
{giuffra,silveira}@inf.ufsc.br
[2] Departamento de Engenharia e Gestão do Conhecimento, Universidade Federal de Santa Catarina (UFSC), Florianópolis, SC, Brasil
marina@egc.ufsc.br

Abstract. The computer-assisted education is increasingly exploited, as well as the use of Learning Management Systems (LMS). LMS are used in distance learning and classroom teaching as teachers and students support tools in the teaching–learning process. Teachers can provide material, do activities and create assessments for students. Nevertheless, this procedure is done in the same way for all the students, regardless of their performance and behavior differences. This work proposes an intelligent tutoring system (ITS) integrated with LMS to provide adaptability to it, using Moodle as a case study, taking into account student performance on tasks and activities proposed by the teacher and student access on resources.

Keywords: Learning management systems, intelligent tutoring system, multi-agent.

1 Introduction

LMS are defined as learning interactive tools where the content is available online. They allow the teacher to provide feedback to students in learning activities and are considered important resources for education. [1]

LMS are used satisfactorily in e-learning, however, as a rule, they do not operate in an interactive and personalized way with students in providing study materials and tasks. They provide the same pedagogical mediation resources and the same content for all of them, without considering their specific needs. Currently, many researches are made to incorporate features that take into account the individual characteristics of students. [12]

In order to provide adaptability to learning environments, according to student characteristics, and to allow a greater interactivity degree between the learning environment and the users, the research points to the use of resources provided by artificial intelligence (AI) and in particular the use of multi-agent system-based architectures [17].

In agreement with this emerges the motivation of this research: to enhance the teaching–learning process in LMS using artificial intelligence techniques to make the

J.M. Corchado et al. (Eds.): PAAMS 2013 Workshops, CCIS 365, pp. 316–327, 2013.

LMS more adaptive and more interactive. This paper proposes the use of agent-based ITS architectures to get personalized teaching strategies, taking into account the student performance, exploring their skills, in order to have better and more effective learning in an intelligent learning environment.

This paper is structured as follows: the second section presents the theoretical reference, the third section presents the definition of the model, the fourth section presents an explanation about the model implementation and the last section presents the conclusions.

2 Background

LMS are technological tools and resources using cyberspace to lead content and enable pedagogical mediation through the interaction between the educational process actors [15]. With the advance of technology the use of these environments has increased because of the ease of providing interaction between student and teacher and the ease of access the content from anywhere and at any time.

For Dillenbourg [8], LMS are not only restricted to distance learning. Web-based education is often associated with distance learning; however, in practice it is also widely used to support classroom learning as a teacher's tool to provide materials, to review tasks, to keep track of the students on course (activity logs) and also to evaluate them. For students, the environment facilitates the delivery of tasks, the obtaining of materials for the course and the monitoring of their evaluation.

LMS can be enhanced with artificial intelligence techniques using cooperative intelligent agents (working in background) or animated pedagogical (interacting with the user). In the cooperative case, a multi-agent modeling is done, where each agent has a specific role and communicates with other agents. These agents are not visible to the user. In the case of pedagogical agents multimedia resources to create an animated character who interacts with the student are used [13]. Resulting in intelligent learning environments.

An agent is a cognitive entity or an abstraction of a device that can perceive its environment through sensors and can act upon that environment through actuators. A human agent has eyes, ears and other parts as sensors, and hands, legs, mouth, and other body parts such as actuators. An agent robot has infrared cameras and locators as sensors and various motors as actuators. A software agent has encoded bit strings as perceptions and actions. [18]

To Wooldridge (2009), an agent is a computer system situated in some environment, capable of perform autonomous actions in order to meet the goals that are delegated to it. In this context, autonomy is to have the ability to decide how to act in order to achieve the goals.

According to [5], an agent always requires a certain amount of intelligence to accomplish their tasks. An agent without intelligence can be any program of traditional software because it also performs specific tasks. Only intelligence allows an agent to perform tasks largely independently, requiring the user participation only to important decisions.

A rational agent is one who chooses perform actions according to their own interests, given the beliefs he/she has about the world. For example, if someone wants to keep itself dry and has the belief that it is raining, then it is rational to take an umbrella when going home. The B.D.I. (Belief, Desire, Intention) model recognizes the importance of beliefs, desires and intentions in rational actions. [20]

An important aspect of B.D.I. architecture is the notion of commitment to previous decisions. A commitment incorporates the balance between reactivity and direction to a goal of an agents oriented system. In an environment that changes constantly, the commitments give a sense of stability to the agent reasoning process. [16].

B.D.I. agents are rational agents who use the own beliefs that they have about the world to realize their desires, having the intention to perform them. The B.D.I. architecture is usually employed for the construction of so-called pedagogical agents.

Pedagogical agents are those whose goal is to help students in the teaching-learning process. According to Giraffa (1999), incorporate agents in an educational program is to intensify the pedagogical aspects desirable in the environment. Pedagogical agents can be divided into agents directed to goals and agents directed to utilities and have as main properties the autonomy, social ability, proactivity and persistence.

In practice, systems with only one agent are not common. The most common are the cases of agents that inhabit an environment containing other agents. The main focus of multiagent systems is to provide mechanisms to create computer systems from autonomous software entities, called agents, that interact through an environment shared for all the agents of a society. (BORDINI, 2001).

According to Bordini et al. [3], there are two major types of multi-agent systems: reactive and cognitive. The reactive acts under a stimulus-response scheme; the cognitive has, in general, few agents because each agent is a complex and computationally heavy system.

The BDI model represents a cognitive architecture based on mental states, and has its origin in the human practical reasoning model. An architecture based on the BDI model represents its internal processes through the mental states: belief, desire and intention, and defines a control mechanism that selects in a rational way the course of actions [9].

In the context of this work, an agent is considered as an autonomous entity, able to make decisions, respond in a timely manner, pursue goals, interact with other agents, and has reasoning and character. This agent is a of type BDI, with beliefs, desires and intentions, and operates in a LMS as an intelligent tutor.

ITS are complex systems involving several different types of expertise: subject knowledge, knowledge of the student's knowledge, and pedagogical knowledge, among others. According to Santos et al. [19], an ITS is characterized for incorporating AI techniques into a development project and acts as a helper in the teaching–learning process.

According to Conati [7], ITS are an interdisciplinary field that investigates how elaborate educational systems provide adapted instructions to the needs of students, as many teachers do.

ITS research has been investigating how to make computer-based tutors more flexible, autonomous and adaptive to the needs of each student by giving them explicit

knowledge of the relevant components of the teaching process and reasoning skills to convert this knowledge into intelligent behavior.

To Giraffa and Viccari [11], ITS developments consider a cooperative approach between student and system. Research in ITS is concerned about the construction of environments that enable more efficient learning. [10].

ITS offer flexibility in the presentation of material and have the major ability to respond to students' needs. They seek, in addition to teaching, learning relevant information about the student, providing an individualized learning. ITS have been shown to be highly effective in improving performance and motivation of students [14].

ITS in LMS potentiate the teaching–learning process, making the virtual environment into an intelligent learning environment. Intelligent learning environments use AI techniques to respond to students' needs, making that learning personalized [14].

An intelligent learning environment is a kind of intelligent educational system that combines features of the intelligent tutor traditional system with learning environments [6].

According [17], the intelligent learning environment must build and update the student model in terms of what he/she already knows, which can vary significantly from one student to another.

3 Definition of the Model

The aim of this work is to create an agent architecture and an agent knowledge base that compose an ITS with information obtained from database of a teaching–learning virtual environment. For this, a case study is done based on the Moodle platform architecture, chosen because it is a platform widely used today, in addition to being consolidated from the standpoint of operation, and also to be formally used in the institution where the research is performed.

The classical model of ITS contains the pedagogical model, the student model, the domain base and the control. In the proposed model, two types of agents, called "Bedel" and "tutor" are used. The Bedel Agent and all their knowledge and interaction structure corresponds in the classical model of intelligent tutoring to the Pedagogical Model. The Tutor Agent and all their structure corresponds in the model of intelligent tutors to the Student Model. The content of the discipline, in turn, may be associated with the abstraction of the Domain Base. These correlations are shown in Figure 1.

In the proposed model the teacher chooses the Pedagogical Model to be used in the course, resources (readings, articles, videos) and activities (papers and exams) that he/she considers necessary to exploit optimally all course features during the semester, and configures the Bedel Agent according to this model, through the LMS interface. This agent helps the teacher. It is the tutor of the course.

The Tutor Agent is the agent that has contact with the student. It guides the student, indicating changes in performance, each time an activity is evaluated, encouraging him/her to improve when he/she has had a drop in performance or congratulating him/her when he/she has had a better performance. These guidelines are sent to students through the LMS messages.

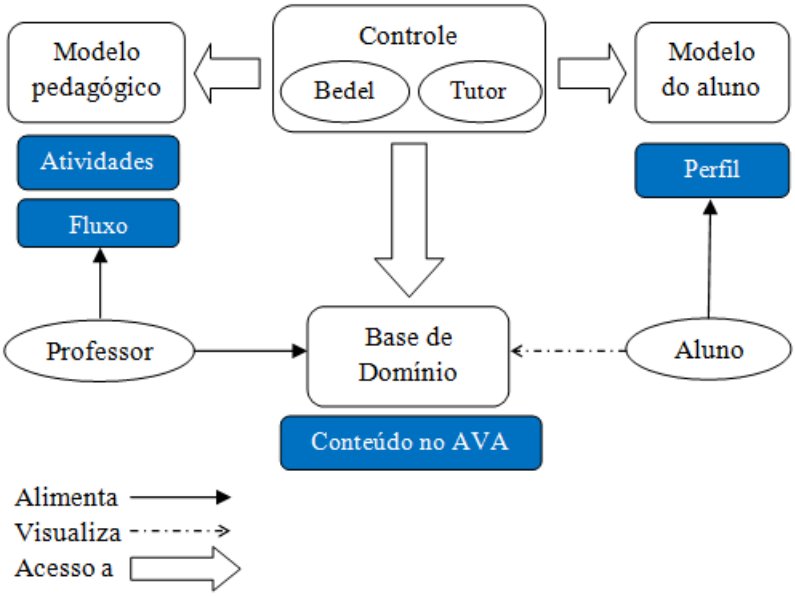

Fig. 1. Classic model with proposed model

The proposed model considers an LMS with a large amount of students, teachers and disciplines, predicting therefore the existence of an Bedel agent for each discipline and an Tutor agent for each student, being that this Tutor agent checks the student's performance in all disciplines in which it is enrolled.

Thus, the model proposes a multi-agent system, with several Tutor agents that communicates with the Bedel agent of each discipline, and Bedel agents providing activities and resources for all discipline students in a personalized way.

This scenario, dynamic and complex, demands the need of independent and autonomous agents that collaborate with each other to achieve the defined purposes, globally, given the directives proposed by each teacher in each discipline, in developing the pedagogical model and the model domain of each one of them.

The system model contains the LMS, the actors (student and teacher, users of the LMS), Bedel and Tutor Agents and the different interactions between them. The teacher adds the resources and activities on the LMS and configure them for the Bedel Agent. After that, the Bedel updates the LMS database to show the resources and activities to the students in a personalized way.

Each time the student accesses the LMS, the Tutor Agent checks the information existing on database about that student and updates the belief it has about him/her. Thereafter, the Tutor agent sends messages to the student using the LMS messaging feature.

The Bedel is the course agent. It constantly checks if any task was evaluated by the teacher, it calculates the students performance and inserts on database the information about their grades and the information required for the LMS to show new activities and resources to the students, taking into account their activity grades and their interaction in different resources of the course.

The database has information concerning the student, such as personal data, performance data and data from student interaction on the system. Every student interaction in the environment is saved in a log on database. Similarly, the student performance in each activity and task is stored on database and is updated every time the student access and interact in the environment, providing rich material for the agent's performance.

In the proposed model, the student model is represented by the student id, his/her grades in the different activities proposed by the teacher and the log information of these activities.

The agents (Bedel and Tutor) access the database as often as necessary to update the course and students information. They share the database information. The Tutor Agent updates the belief that it has about the student and if required, shows a message for him/her about his/her performance. The Bedel Agent gets from database, the configuration of the resources and activities of the course, made by the teacher, moreover, configures their display for each student profile and checks if new tasks were evaluated to send a message to the Tutor Agent with the information of the student to update the belief it has about the student profile data.

In the proposed model, formed by the actors teacher and student and the intelligent tutors Tutor and Bedel, students are grouped into three different profiles, according to their performance (grades) in the tasks and their access on different resources (study materials). These profiles are basic, intermediate and advanced.

This separation into groups takes into account the "grade profile" of each student, which is calculated as follows: The student grade of the last activity assessed by the teacher and the grade of the student access on reading that is a prerequisite for activity are summed. If the student accesses the reading, two points in the activity grade are added. If he/she does not access, four points are added. This difference is given for increasing the possibility to the student who does not access the reading, to have a higher profile grade and then go to a higher level task than the profile that he/she would belong to if he/she had a lower grade, stimulating him/her to read before accomplishing future activities.

After this, the average value of the profile grade field is computed for all students. The lowest value considered for the profile average is 6. The maximum value considered for the profile average is 8.

The student belongs to the average profile if his/her profile grade is 0.5 less or more than the average profile grade in his/her class; for example, if the average profile grade is 7.5, he/she will be in the intermediate profile if he/she has a profile grade between 7 and 8. The student who has a higher grade with more than 0.5 of difference with the average will be in the advanced profile and the student who has a lower grade with more than 0.5 of difference will be in the basic profile.

The Bedel is the course agent. It receives the information entered by the teacher when configuring the tutoring system. With this information, the Bedel knows what behavior it must follow to display the course in a personalized way according to each student profile by inserting the required information on database. Also, the Bedel checks for teacher updates on worksheet, calculates the profile grade of students, calculates the average profile grade, updates the database with these grades and

communicates with Tutor Agent to inform the performance of the student with who the Tutor Agent is in contact.

The Tutor Agent is the agent that communicates with the student, it gets information sent by Bedel Agent about the student performance, each time the worksheet is updated and communicates with the student to encourage him/her and congratulate him/her according to what he/she needs at the moment.

The proposed model has the next steps:

Step 1:

- The teacher adds the resources and activities normally in the LMS, using the LMS tools as usual.
- The teacher sets the Bedel Agent using the environment interface including:
  - Level of difficulty (basic, intermediate, advanced and general) for each resource and activity that the teacher added previously in the LMS. The "general" difficulty level is selected when the teacher wants to show some resource or activity to all students equally.
  - The resources and activities where the students should start the process. The first reading (resource) and the first activity are shown for all students and the teacher have to indicate which ones they are.
  - The dependencies between the activities and resources.
- With this information the Bedel Agent knows the resources and activities of the course and knows how the course should be developed for each type of student, according with his/her profile.
- The teacher see the dependency graph generated by the system after the completion of the previous step.

Step 2:

- The teacher assigns grades in the proposed activities, after being performed by the students, and updating the worksheet for all students in the course.
- The Bedel Agent checks that were given grades for all students in a particular activity and calculates the student profile grade using the grade given by the teacher and the access log in the resources flagged as prerequisites of the activity, at the initial configuration of the teacher.
- The Bedel Agent calculates the average of the profile grade of all students and the students are separated by the profile into groups (basic, intermediate or advanced) according to their profile grade. Who has the average grade is in the intermediate profile, who is below the average is in the basic profile and who is above the average is in the advanced profile.

Step 3:

- The Bedel Agent checks the profile to which the students belong and shows the following activities according to that profile.
- Students access the activities in a personalized way, according to the grade obtained in previous activities and their access on the resources (readings).

- Each time the Bedel Agent calculates the profile grade it updates the database with the current student profile, which can go from basic to intermediate or advanced and vice versa, during the time the course is offered.
- The process is repeated from step 2.

4 Implementation of the Model

The model integrates concepts of ITS architectures with LMS that have their use consolidated as Moodle, which are not adaptive for itself only, and can be potentiated with artificial intelligence techniques, resulting in intelligent learning environments which are shown to be adaptive and more suitable to the implementation of teaching defiant methodologies for the student.

The use of agents in the implementation of this model is important because of the agent's ability to adapt to environment changes, showing resources and activities to students in a personalized way, according to their performance in the course, and taking into account the teacher's initial settings.

For the agent implementation the Jason tool was used, which is an interpreter for an extended version of AgentSpeak, oriented agent programming language, implemented in Java. The basic idea of AgentSpeak is to define the know-how (knowledge about how to do things) of a program in the form of plans [4].

The teaching–learning virtual environments are designed to enable the knowledge-building process. Different to conventional software, which seeks to facilitate the tasks achievement by user, learning environments incorporate the complexity to more flexible different forms of users (students), relations, to learn content, and to collaborate. These environments are used by students of various cognitive profiles [2].

According to the research of [1], Moodle has a good architecture, implementation, interoperability and internationalization, besides having a very strong community, is free and its accessibility is average. It has almost the highest score in expected functionality of an e-learning platform, and has the best rating in the adaptation category. In addition, Moodle has personalization features and adaptability.

The version of Moodle LMS used for this work is 2.2, where the task condition resource is available. This condition allows the availability of content and activities with a restriction. This feature must be activated by the Moodle administrator in the environment advanced settings, enabling the option "Enable tracking of completion" and "Enable conditional access." Moreover, in course settings, in "student progress" topic, the teacher must enable the completion tracking option.

With this feature enabled, tasks can be made available only to students who perform the defined prerequisites, which can be: a grade on a specific activity; the viewing of a resource; or his/her grade.

In this work, the availability of resources and activities is done taking into account the student performance and his/her access on the system. The resources and activities are made available to the student depending on his/her profile grade, computed according to his/her performance and participation in the course. The information between the agent system and the LMS is exchanged through the database of the

learning environment which contains information about the prerequisites of tasks and resources, defined by the teacher at time of configure the tutor (Fig. 2).

The agents are connected with the LMS through the database. The interface in which the teacher sets the levels and priorities of the resources and tasks in the environment is developed. The database is adapted with the creation of the tables for the profile types, students profile grades, average grades of profiles, availability according to the profile, according to the level of tasks, evaluated tasks, and dependencies of resources and activities set by the teacher, and the integration of the agents actions with the Moodle LMS is done.

The development, in Moodle and using Jason, has 4 steps.

4.1 Tutor Block Development

It was developed a moodle block for the teacher to configure the agent. This block was developed following the Moodle standard programming for creating blocks, using PHP language and accessing the Moodle database using SQL language.

The name of the block was defined as "Tutor Block". Here the teacher sets the level of the different resources and activities, whether they are basic, intermediate or advanced, moreover, choose the first resource and activity, which are available equally to all students, and are the basis for the initial calculation of the student profile grade, and finally chooses the prerequisites for each activity and resource of the course.

After the teacher configures the agent through the block, he/she can see the dependency graph (Fig. 2) generated after setting all prerequisites.

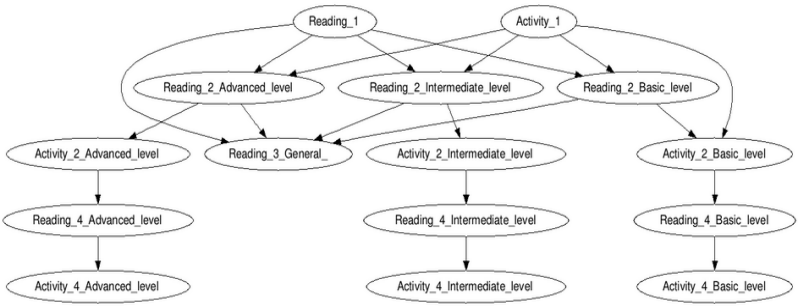

Fig. 2. Dependence graph

4.2 Development of the Calculation of the Student's Profile (Bedel Agent)

The agent programming was performed using the Eclipse IDE, with Jason plugin. The agents were developed with artifacts implemented in Cartago, in the Java language.

The agents' implementation is basically divided into the following files:

- jasonTutor.mas2j: Definition file of the multi-agent system. Here is specified the centralized infrastructure, the Cartago environment and the Tutor and Bedel Agents.
- BD_Artifact.java: File with the artifact code, which makes the connection between the agents and the LMS database.
- tutor.asl: File with the implementation of the Tutor Agent, who is the agent that has contact with the student.
- bedel.asl: File with the implementation of the Bedel Agent, teacher agent who follows the settings defined by the teacher in the LMS block, to show to the students the resources and activities depending on their performance.

4.3 Development of the Availability of Resources and Activities Code (Moodle)

For the availability of resources and activities according to the student profile was created the tutor_profile_availability table in the database, to store the information of the minimum and maximum grades of the intermediate profile. Verifying this information the resources and activities can be provided, according to the students profile.

In addition to this table, the Moodle source code is modified in conditionlib.php method "is_available" by entering the code needed so that resources and activities are shown on the LMS for each student according to their profile, taking the information added by the Bedel Agent into the database, after making the calculation of the grade profile.

At this stage, also, the bedel.asl file is updated, inserting methods that enable communication between it and the Tutor Agent. Sending the student information to him.

4.4 Development of the Tutor Agent - Feedback to the Student

In this step the Tutor Agent code is developed, this implementation is done in the tutor.asl file. This agent is responsible for sending messages of encouragement to the student, depending on his/her performance.

The Tutor Agent has the belief of the student profile grade, which is updated each time the Bedel Agent sends the message with a new profile grade. And according to the student situation the Tutor Agent sends a message to him/her, congratulating him/her or encouraging him/her to improve.

The Tutor Agent must have a LMS user account for sending the messages to the student.

5 Conclusions

In this study is proposed a solution for LMS to assist teachers to provide activities and resources to the students in a personalized way depending on their performance and his/her behavior in the course using an ITS architecture.

Students are assessed by their interaction in the course and the grades obtained in tasks, creating different profiles for groups of students with the same behavior. More advanced tasks are available for students who have improved performance, enabling more efficient learning, exploring student skills, and maintaining a basic level for learning the course content.

Works related to LMS and adaptivity in general differentiate students by learning style – for example, a student who learns better with pictures than with reading lots of text. In this work students are distinguished by their performance, taking into account the grades obtained, and their participation (access) in the various resources available in the course, creating an adaptive environment that constantly updates the profile of students, and therefore, a student with a basic profile, at the end of the course may have an average profile. These profile changes can be studied and displayed to the teacher, in an extension of this model.

The main contribution of this work is to add the advantages of the LMS to intelligent tutors and vice versa, creating an intelligent learning environment.

References

1. Ajlan, A.-A., Husein, Z.: Why Moodle. In: International Workshop on Future Trends of Distributed Computgin System. IEEE (2008)
2. Boff, E.: Collaboration in Learning Intelligent Environments mediated by a Social Agent Probabilistic. Thesis (Ph.D.) Computer Science Course. Federal University of Rio Grande do Sul, Porto Alegre (2008) (in Portuguese)
3. Rafael, B., Renata, V., Moreira Alvaro, F.: Multiagent Systems Fundamentals. In: Ferreira, C.E. (ed.) Informatic Update Day, JAI 2001, vol. 2, ch. 1, pp. 3–44. SBC, Fortaleza (2001) (in Portuguese)
4. Bordini, R.H., Hubner, J.F., Wooldridge, M.: Programming Multi-Agent Systems in AgentSpeak using Jason. Editora Wiley, England (2007)
5. Walter, B., Rüdiger, Z., Hartmut, W.: Intelligent Software Agents – Foundations and Applications. Springer, Heidelberg (1998)
6. Peter, B.: Student model centered architecture for intelligent learning environment. Proceedings of Fourth International Conference on User Modeling (1994), Diponível em: http://www2.sis.pitt.edu/~peterb/papers/UM94.html (Acesso em Dez. 04, 2011)
7. Conati, C.: Intelligent Tutoring Systems: New Challenges and Directions. Paper Presented at the Proceedings of the 21st International Joint Conference on Artificial Intelligence (2009)
8. Dillenbourg, P.: Virtual Learning Environment. In: EUN Conference 2000. Workshop on Virtual Learning Environment (2000), http://tecfa.unige.ch/tecfa/publicat/dil-papers-2/Dil.7.5.18.pdf
9. Fagundes, M.: An environment for development of BDI agents. Course Completion Work. Federal University of Pelotas (2004) (in Portuguese), http://www.inf.ufsc.br/~silveira/INE602200/Artigos/TCC_Moser.pdf
10. Frigo, L.B., Pozzebon, E., Bittencourt, G.: The Role of Intelligent Agents in Intelligent Tutoring Systems. In: Proceedings of the WCETE - World Congress on Engineering and Technology Education, São Paulo, Brasil, pp. 667–671 (2004) (in Portuguese)

11. Giraffa, L.M.M., Viccari, R.M.: The Use of Agents Techniques on Intelligent Tutoring Systems. In: Computer Science SCCC 1998. XVIII International Conference of the Chilean Society, pp. 76–83. IEEE Computer Society, Washington, DC (1998)
12. Graf, S., Kinshuk: Analysing the Behaviour of Students in Learning Management Systems with Respect to Learning Styles. In: Wallace, M., Angelides, M.C., Mylonas, P. (eds.) Advances in Semantic Media Adaptation and Personalization. SCI, vol. 93, pp. 53–73. Springer, Heidelberg (2008)
13. Jaques, P.A., Vicari, R.M.: State of the Art in Intelligent Learning Environments that consider the student affection. Computers in Education 8(1), 15–38 (2005) (in Portuguese)
14. Lima, R.D., Rosatelli, M.C.: An intelligent tutoring system to a virtual environment for teaching and learning. In: IX Workshop de Informática na Escola, Campinas. Anais do XXIII Congresso da Sociedade Brasileira de Computação (2003) (in Portuguese)
15. Pereira, A.T.C., Schmitt, V., Álvares, M.R.C.: Virtual Learing Environments. Culture Bookstore (2007) (in Portuguese), http://www.livrariacultura.com.br/imagem/capitulo/2259532.pdf
16. Rao Anand, S., Georgeff Michael, P.: Bdi Agents: From Theory to Practice. In: Proceedings of the First International Conference of Multiagents Systems. AAAI (1995), Disponível em: https://www.aaai.org/Papers/ICMAS/1995/ICMAS95-042.pdf (Acesso em: Dez. 04, 2011)
17. Silveira, R.A.: Intelligent Distributed Learning Environments. CPGCC da UFRGS, Porto Alegre (1998) (in Portuguese)
18. Russell, S., Norvig, P.: Artificial Intelligence: A Modern Approach. Prentice-Hall, Inc., New Jersey (2002)
19. dos Santos, C.T., Frozza, R., Dhamer, A., Gaspary, L.P.: DÓRIS - Pedagogical Agent in Intelligent Tutoring Systems. In: Cerri, S.A., Gouardéres, G., Paraguaçu, F. (eds.) ITS 2002. LNCS, vol. 2363, pp. 91–104. Springer, Heidelberg (2002)
20. Wooldridge, M.: Reasoning about Rational Agents. The MIT Press, Cambridge (2000)
21. Wooldridge, M.: An Introduction to Multiagent Systems, 2nd edn. John Wiley & Sons Ltd., Hoboken (2009)