Adaptativity Supported by Neural Networks in Web-based Educational Systems

Fabiano DORÇA
UNIPAM- Centro Universitário de Patos de Minas
FACIA- Faculdade de Ciências Administrativas
Rua Major Gote 808– 38702-054 – Patos de Minas/MG – Brasil
Phone: +55 (034) 3823-0300
E-mail: fabiano@unipam.edu.br

and

Carlos LOPES, Márcia FERNANDES, Robson LOPES
Universidade Federal de Uberlândia
Faculdade de Computação
Av. João Naves de Ávila, 2160 – 38400-902 – Uberlândia/MG – Brasil
Phone: +55 (034) 3239-4144
E-mail: {robson, marcia, crlopes}@ufu.br

ABSTRACT

Effective evaluation and curriculum sequencing are complex activities in distance learning systems. These activities are closely related since a curriculum sequence should take student performance into account. In curriculum sequencing many features should be considered such as student preferences and abilities, relationship among the concepts, previous knowledge, and learning goals. A successful system take these features into account and should be able to dynamically change a curriculum sequence when values associated to those features change during the learning process. In this paper we describe a multiagent system that achieves this adaptivity by the use of a neural network.

1. INTRODUCTION

In this paper we describe a multiagent system for web-based distance education (WBDE) with characteristics of adaptivity and intelligence. The adaptivity mechanism is responsible for making the system capable of changing its own characteristics automatically by for a particular student. These modifications are based on how the student interacts with the system and the degree of knowledge acquired. In our system adaptivity is achieved by the use of a neural network, which allows changes in the curriculum sequencing during the learning process [1].

Our approach is based on Intelligent Tutoring Systems (ITS) and Multi-Agent Systems (MAS) to reach the characteristics of adaptivity and intelligence. The system developed consists of two main parts, the Course Management System (CMS) and the Multiagent Intelligent System (MIS), which are discussed in Section 2. In Section 3 we describe the process of adaptivity using neural nets. In Section 4 we show some implementation issues. Finally, Section 5 concludes the paper.

2. THE PROPOSED ARCHITECTURE

The proposed architecture consists of two main parts, the Course Management System (CMS) and the Multiagent Intelligent System (MIS), discussed bellow.

2.1 – The CMS Component

The CMS component is responsible for interacting directly with the student. It allows the capture of actions the student performs when interacting with the system. Its main functions include presenting the contents of a course and supplying the necessary infrastructure, as for example, some communication tools that provides the interaction among students and tutors. It also supplies all the necessary tools for user and course management and implements an authoring tool for the content generation.

Two environments make up the Course Management System: the Course Environment and the Administrative Environment. The Course Environment interacts directly with the students. Its main function is to present the course content and provide communication tools. The Administrative Environment has all the necessary procedures for user management, course management, tests and content generation.

2.2 – The MIS Component

The MIS depicted in Figure 1 provides adaptivity and intelligence to the CMS through the introduction of intelligent agents. This component is responsible for all whole pedagogical processing including curriculum sequencing, diagnosis of learning problems, knowledge evaluation, behavior evaluation and help to the students. These are high complex tasks and usually demand a certain degree of concurrency, continuous execution, decentralization of the information ownership, decentralization of task execution involved in the process, autonomy and cooperation between the agents. In this context a MAS application has great value [2].

The MIS component implements all the modules that make up an ITS - the Knowledge Module (Knowledge Domain Database), the Pedagogical Module (Pedagogical Database, Pedagogical Agent and Evaluation Agent), Expert Module (Expert Agent), Communication Module (Assistant Agent and CMS) and the Student Model (Student Model Database). The MIS structure is depicted in Figure 1.

According to Figure 1 the Assistant Agent receives information from the student (1) and sends it to Evaluation Agent (2). In order to evaluate a problem solved by the student, evaluation agent can request help from the Expert Agent (3).

As a result of this interaction, the Student Model is updated (4) and this is communicated to the Pedagogical Agent (5). Based on the Student Model, the Pedagogical Agent can update the student curriculum (6) with appropriate content (7). This process was accomplished by instructional planning techniques and is discussed in details in [3].

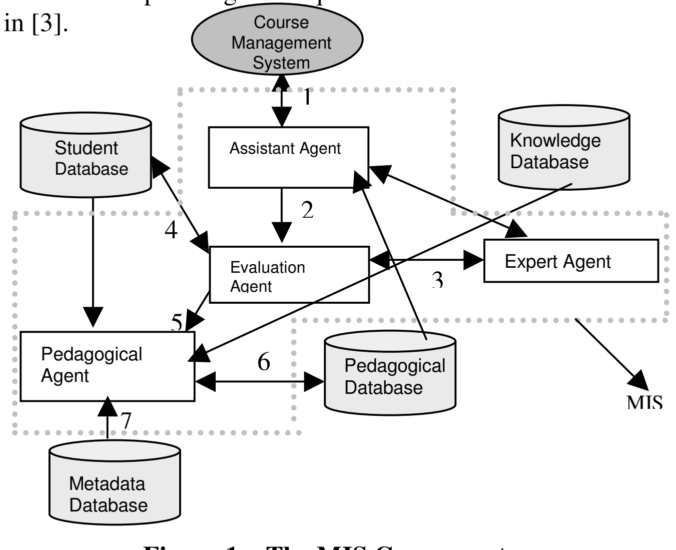


Figure 1 – The MIS Component

Our approach takes into account a high specialized division of the tasks involved in the teaching-learning process in order to improve the efficiency of the message passing, minimizing the traffic and taking the maximum advantage of the distributed processing provided by MAS.

Following we present some important information about Pedagogical Agent and Evaluation Agent. More details about the CMS and MIS components can be found in [3], [4] and [5].

2.2.1 – Pedagogical Agent

In order to implement the Pedagogical Agent, an efficient knowledge representation was necessary. Our proposal for the knowledge representation was inspired by ABITS [6]. In this proposal, the knowledge module is composed of two parts, the Knowledge Database and the Metadata Database.

The Knowledge Database contains all the types of instructional material that can be presented to the student. This database may be centered in one server or spread accross some servers over the Web. In this way, the Knowledge Database includes the course contents stored in formats that can be visualized with a web browser. In addition, the Metadata Database stores metadata information about the contents in the Knowledge Database.

The Metadata Database contains the information necessary to index the instructional material, attributing meanings to them and setting the dependences between them. These information will be used in the adaptation of the curriculum to be presented the a particular student.

The Pedagogical Agent needs pedagogical information such as pedagogical type, interactivity type, interactivity level, semantic density, educational environment, duration, dificulty level, user age and learning time. So, we use the LTSC LOM (Learning Object Metadata) from IEEE [7].

In the Metadata Database the Learning Objects represent instructional material that can be presented through the Web, as for example a lesson (HTML page), a simulation (Java applet), a virtual world (VRML archive), a test (HTML page with an evaluation form) and other types of objects supported by the Web. A Learning Object must define the minimum set of properties necessary to allow the management, localization and the evaluation of the instructional material represented by the object.

The adoption of the IEEE LTSC LOM was based on the necessity of well accepted standard that allows the capacity to reuse materials available on the Internet. Another important characteristic of this standard is the capacity to store a large variety of pedagogical information, making possible the implementation of efficient pedagogical processing.

The curriculum sequencing, which is the main activity of the pedagogical agent, is achieved by using a content planner and a presentation planner. The content planner is responsible for defining and ordering the contents to be presented to the student. Our approach is based on hierarchical planning [8] and defines abstract and primitive operators. Also, an important feature in our work is that events are considered during planning time. After the generation of the content plan, a presentation plan is achieved. The presentation planner, based on the student model, decides how to present the content to the student. More details about the curriculum sequencing can be found in [3].

2.2.2 - The Evaluation Agent

The Evaluation Agent is responsible for evaluating the progress of the student during the course. As a result of this process the Student Model is updated.

It has been noted that the Student Model is a very important ITS module because it provides information for the execution of all the functions previously mentioned (pedagogical processing, curriculum sequencing, diagnosis of learning problems, knowledge evaluation, behavior evaluation and help to the student) [9].

The Student Model present in MIS is based on the Overlay Model [9] and uses Fuzzy Sets to represent the degree of knowledge acquired by the student in a certain time. The Fuzzy Sets are characterized mainly by flexibility, efficiency and relative simplicity of implementation when compared with other techniques. In addition, they achieve satisfactory efficiency. The evaluation algorithm is divided in two parts: the Knowledge Evaluation (KE) and the Behavior Evaluation (BE).

Both BE and KE use Fuzzy Rules to evaluate the student. Initially, our system possesses three input linguistic variables, used by KE: **Answer Quality**, **Answer time** and **Behavior** (calculated by BE), and the resulting variable, **Performance**. Based on these linguistic variables, an example of one of these Fuzzy Rules is given by:

IF the **Answer** is *Corrrect*, the **Answer time** is *Reasonable* and the **Behaviour** is *Bad* THEN the **Performance** is *Satisfactory*.

The linguistic terms to **Answer** are *Wrong, More or less correct, or Correct*. **The Time of Reply** may be *low, reasonable or high*. The liguistic terms to **Behavior** are *Bad, Normal and Good*. Finally, the linguistic terms to the **Performance** are *Low, Regular, Satisfactory, Good, Excellent*. This is obtained as a long-term evaluation, at the end of a chapter. So far, the KE has 25 fuzzy rules.

In the student behavior evaluation, the following questions must be answered: What characterizes the student behavior? What could modify the student behavior? Therefore, the input linguistic variables used by BE are **Time spent on a page; Scroll bar movement** and the **Participation in Chat and Forum.** The output variable is **Behavior**, as explained above. This is obtained as a long-term evaluation.

The liguistic terms to **Time spend in a page** are *Fast, Normal, Slow.* **The Scroll bar movement** may have the values *Low, Normal High.* The **Participation in Chat and Forum** variable may be *Bad, Resonable or Good.* The BE has 27 Fuzzy Rules.

In order to transform these linguist variables into numeric values, some membership functions map the input variables in values inside the interval [0, 1]. These are the values to be stored in the Student Model for each learning section (or unit) in the curriculum of the student. The neural net presented in next section uses these values to classify the student, allowing the pedagogical agent to adapt his/her curriculum sequencing.

In addition, the Student Model stores the student learning preferences and personal characteristics so that the most adequate Learning Objects can be introduced into the student curriculum.

3. A PROPOSAL OF ADAPTIVITY SUPPORTED BY NEURAL NETS

It is well known that there are many different types of artificial neural nets. Among them, the LVQ (Learning Vector Quantization) nets allow implementing the pattern classification, where each output unit represents a particular class. LVQ uses a supervised training method, which requires known target output classifications for each input pattern [10].

What we propose here is the use of a LVQ net to classify the student so the pedagogical agent can properly adapt his/her curriculum. In our work, we define five different types of students that represent the patterns we want to classify using a LVQ neural net. Each one of these patterns has different pedagogical characteristics. So, we propose a neural net able to constantly analyze the student model and classify the student in an appropriate pedagogical class. This is a short-term evaluation, obtained after each unit inside a chapter. This classification allows the pedagogical agent to act according to the student necessities, updating its curriculum sequencing in order to diagnose its learning problems. This process brings the characteristic of adaptivity, which is the main goal of a ITS.

To execute the student classification, the neural net takes some student characteristics (in the student model) during a specific unit or learning section:

- **Answer:** evaluation of the answers in the exercises
- **Time of Reply**: Time spent to solve the exercises.
- **Time spend in a page:** Time spent to study the pages.
- **Scroll bar movement:** Number of page rollbacks related to a page content.
- **Participation in Chat and Forum:** Number of participation in these tools.

For each of the learning objects the human tutor must define the values that are considered to be good, or normal. For example, for a specific learning object the tutor could define:

- **Time spend in a page:** 5 minutes
- **Scroll bar movement:** 10 rollbacks
- **Participation in Chat and Forum:** 5 times

For an evaluation object, the tutor could define:

- **Answer:** 70% of correctness
- **Time of Reply**: 30 minutes

At the end of a learning section, the values obtained by the student are sent to evaluation agent, which is responsible for calculating the fuzzy values used by the neural net.

This agent uses the parameters defined by the tutor for the learning objects and generates values in the interval [0, 1] for each one of the five parameters described above. Values close to 1 show that the student obtained a high value for that parameter, which is above that considered as normal. Values close to 0 imply that the student obtained a low value, which is below that considered as normal.

According to these patterns and the values obtained by the student, the neural net classifies the student in one of the following

1. Too difficult
2. Difficult
3. Normal
4. Easy
5. Too easy

In distance learning environments students classified either in class 1 or in class 5 can easily lose their motivation and abandon the course. In this context, adaptability is a very important characteristic as it increases the effectiveness of a distance learning environment. After classifying the student, the pedagogical agent updates its curriculum sequencing with specific learning objects that will help diagnose learning problems. The human tutor must have previously created these learning objects. In this approach, we use the LTSC LOM (Learning Object Metadata) proposed by IEEE [7].

3.1 – Topology of the proposed net

The LVQ neural net proposed here has 5 input units, where we put the evaluation obtained by the student in each one of the parameters described above. The net has 5 output units to classify the student in one of the 5 classed defined above. The implementation of the net algorithms was done according to [10]. Figure 2 shows the topology of this neural net.

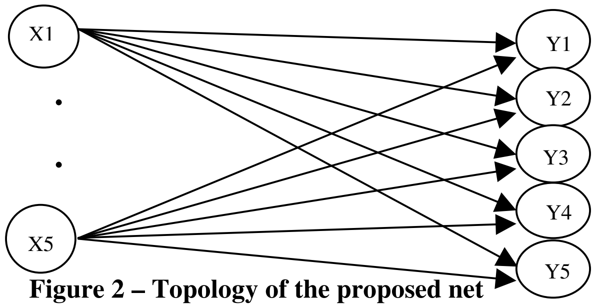


Figure 2 – Topology of the proposed net

According to Figure 2 we have:
X1 = Answer
X2 = Time of Reply
X3 = Time spend in a page
X4 = Scroll bar movement
X5 = Participation in Chat and Forum
Y1 = class 1
Y2 = class 2
Y3 = class 3
Y4 = class 4
Y5 = class 5

3.2 – Training the net

Training the net correctly is very important to obtain good results (in our case, correct classifications of the students). The previously constructed training pairs used here were obtained with the help of rules. Consider the following:
A = **Answer**
B = **Time of Reply**
C = **Time spend in a page**
D = **Scroll bar movement**
E = **Participation in Chat and Forum**
"high" = to represent values close to 1
"low" = to represent values close to 0
"medium" = to represent values close to 0.5

The following rules were defined:

1. (A) high, (B) low, (C) low, (D) low, (E) low➔ too easy – class 5
2. (A) high, (B) medium, (C) low, (D) low, (E) low➔ easy – class 4
3. (A) medium, (B) medium, (C) medium, (D) medium, (E) medium ➔ normal – class 3
4. (A) medium, (B) high, (C) high, (D) high, (E) high ➔ difficult – class 2
5. (A) low, (B) high, (C) high, (D) high, (E) high➔ too difficult – class 1

Using these rules we defined the training pairs that are specified in Table 1.

Table 1 – Training Pairs

<u>**A**</u>	<u>**B**</u>	<u>**C**</u>	<u>**D**</u>	<u>**E**</u>	<u>**Class**</u>
1	*0*	*0*	*0*	*0*	*5*
0,9	*0,1*	*0,1*	*0,1*	*0,1*	*5*
0,8	*0,2*	*0,2*	*0,2*	*0,2*	*5*
1	*0,5*	*0*	*0*	*0*	*4*
0,9	*0,5*	*0,1*	*0,1*	*0,1*	*4*
0,8	*0,5*	*0,2*	*0,2*	*0,2*	*4*
0,5	*0,5*	*0,5*	*0,5*	*0,5*	*3*
0,5	*1*	*1*	*1*	*1*	*2*
0	*1*	*1*	*1*	*1*	*1*
0,1	*0,9*	*0,9*	*0,9*	*0,9*	*1*
0,2	*0,9*	*0,9*	*0,9*	*0,9*	*1*

The definition of these training pairs was done during tests supported by a prototype. During the tests, many adjustments were done in the training pairs in order to make the classification process more effective.

The output of the net is a target vector that represents the student classification. Each target vector has only a single digit 1. The rest of its elements are made up by digits 0.. The digit 1 tells the proper classification of the associated input. For example, for a student classification equals to 4, we have the output [0 0 0 1 0].

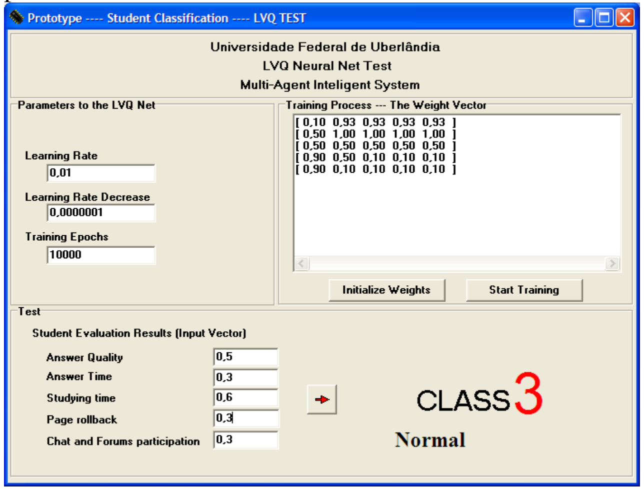
3.3 – Testing the net

Figure 3 presents the prototype during a test. The prototype allows the user to inform the net parameters (learning rate, learning rate decrease, training epochs). For this test, we used 0,01 for the learning rate, 0,0000001 for the learning rate decrease and 10000 training epochs.

After executing the training, the user may enter the student evaluation results (input vector) and achieve a classification for that student. According to the net, as showed in Figure 3, we have the following input vector:

Answer = 0,5
Time of Reply= 0,3
Time spend in a page= 0,6
Scroll bar movement = 0,3
Participation in Chat and Forums = 0,3

The results in class 3 classify the student learning process as normal.



Prototype ---- Student Classification ---- LVQ TEST

Universidade Federal de Uberlândia
LVQ Neural Net Test
Multi-Agent Inteligent System

Parameters to the LVQ Net
Learning Rate: 0,01
Learning Rate Decrease: 0,0000001
Training Epochs: 10000

Training Process --- The Weight Vector
```
[ 0,10  0,93  0,93  0,93  0,93 ]
[ 0,50  1,00  1,00  1,00  1,00 ]
[ 0,50  0,50  0,50  0,50  0,50 ]
[ 0,90  0,50  0,10  0,10  0,10 ]
[ 0,90  0,10  0,10  0,10  0,10 ]
```

Test
Student Evaluation Results (Input Vector)

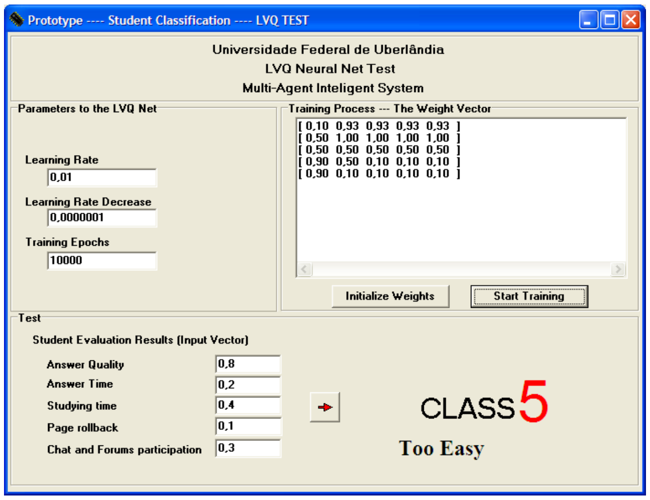
Answer Quality	0,5
Answer Time	0,3
Studying time	0,6
Page rollback	0,3
Chat and Forums participation	0,3

CLASS 3
Normal

Figure 3 – Testing the net

Another test parameters presented bellow:

Answer = 0,8
Time of Reply= 0,2
Time spend in a page= 0,4
Scroll bar movement = 0,1
Participation in Chat and Forums = 0,3



Prototype ---- Student Classification ---- LVQ TEST

Universidade Federal de Uberlândia
LVQ Neural Net Test
Multi-Agent Inteligent System

Parameters to the LVQ Net
Learning Rate: 0,01
Learning Rate Decrease: 0,0000001
Training Epochs: 10000

Training Process --- The Weight Vector
```
[ 0,10  0,93  0,93  0,93  0,93 ]
[ 0,50  1,00  1,00  1,00  1,00 ]
[ 0,50  0,50  0,50  0,50  0,50 ]
[ 0,90  0,50  0,10  0,10  0,10 ]
[ 0,90  0,10  0,10  0,10  0,10 ]
```

Test
Student Evaluation Results (Input Vector)

Answer Quality	0,8
Answer Time	0,2
Studying time	0,4
Page rollback	0,1
Chat and Forums participation	0,3

CLASS 5
Too Easy

Figure 4 – Testing the net

For these input data the network output is class 5 indicating that the learning process is too easy, as shown in Figure 4.

With the final training pairs showed in Table 1, the net gave us coherent answers in all the tests done during this phase.

After this step, the neural net was implanted in the pedagogical agent, which could be tested acting together with the rest of the system, presented in Section 2. Although the tests of the whole system were done with such a small number of learning objects, we could see that this approach is valid and may bring advances in the context of adaptivity in distance learning systems.

4. IMPLEMENTATION ISSUES

The CMS is a common Web application, which consists of a set of scripts that provides dynamic pages to the students. The CMS was implemented using PHP[4]. Therefore, the application is accessible by any Web browser in any computational platform that offers an Internet connection. Moreover, the CMS implementation provides asynchronous interaction for students, without determining the time for the educational activities. The system databases have been implemented using MySQL [4].

The MIS has been developed using Jade Multiagent Platform [11]. Jade complies with FIPA standards for intelligent agents [12], which is the most complete and frequently used specification for multiagent system development. It specifies all the protocols, resources and services that must be provided and how they must be used in the MAS context (for example, name services, yellow pages and communication protocols).

The agents shown in Figure 1 are started when the system is initiated. Although the MIS and CMS make up a unique application, they can run on different hosts. It is possible because MySQL allows remote access, as shown in Figure 5 where H1, H2, H3, H4, H5 and H6 represent different hosts.

Communication between the CMS and MIS has been implemented by means of blackboard software [13], which is made up by two distinct tables inside the system database. One of them stores the messages containing all the information collected by CMS about the user and the other stores the messages containing all the feedback to be presented to the student (generated by the MIS).

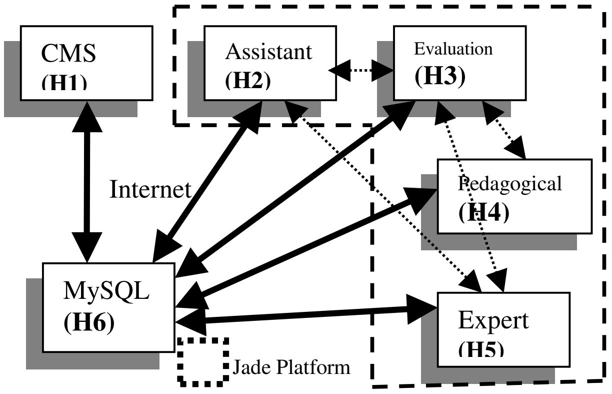


Figure 5 – The Distributed System Components

Only one agent of each type is started in the platform. It is not necessary to have one MIS dedicated to each student because all the messages exchanged between the CMS and MIS and among MIS agents require the student and course identification numbers. This is sufficient to make the agents work well for many students. As the number of students grows it may be necessary to start other agents so they can distribute tasks and improve the time processing.

5. CONCLUSION

Our system presents innovative characteristics in relation to the adaptation of the course for students and evaluation of the teaching/learning process. Intelligent agents have been suggested as a promising application in the attempt of extending Web-based educational systems.

Our approach uses neural nets and instructional planning to obtain the characteristic of adaptivity. In this way, the application of neural nets in classifying students contributes to the developing of Intelligent Tutoring Systems. LVQ nets are able to deal with non-linear problems, being possibly applied in solving many other problems in the learning systems. A detailed explanation about neural nets can be found in [6].

This paper focused on the application of neural nets on the curriculum adaptation issues. The other questions involved in the whole process can be seen in [3], [4], [5] and in others works of our research group.

A difficulty found in this work was to define the training pairs and the rules that supported them. It is well known that an improper training process results on an incoherent net that provides incoherent answers.

Some related works are [1], [6] and [14]. This system has not been tested with students. The test results will be shown on a future work.

We believe that the use of multi-agent paradigm to implement ITS's is a very good approach. In addition, we suggest the use of fuzzy sets and instructional planning supported by neural nets to obtain the characteristic of adaptivity, which is present on a ITS.

6. REFERENCES

[1] L. A. M. Zaina, G. Bressan, R. M. Silveira, I. Stiubiener, W. V. Ruggiero, "Analysis and Comparison of Distance Education Environments" - International Conference on Engineering Education August 6 – 10, Oslo, Norway, 2001.

[2] N. R. Jennings, P. Faratin, M. J. Johnson, P. O'Brien, M. E. Wiegand, "Using Intelligent Agents to Manage Business Processes", Proceedings of First International Conference on The Practical Application of Intelligent Agents and Mult-Agent Technology (PAAM96), London, UK, 345-360, 1996.

[3] B. Queiroz. "Automatic Content Generation in a Web Based Education System", Master Thesis, Universidade Federal de Uberlândia, Brazil, 2003 (In Portuguese).

[4] F. Dorça. "A Web-Based Intelligent Multiagent Educational System", Master Thesis, Universidade Federal de Uberlândia, Brazil, 2003 (In Portuguese).

[5] F. Dorça, M. A. Fernandes, C. R. Lopes, "A Web-Based Multiagent Educational System", International Conference on Education and Information Systems, Technologies and Applications (EISTA 2004) and the International Conference on Cybernetics and Information Technologies, Systems and Applications (CITSA 2004), in Orlando, USA, in July 21-25, 2004.

[6] N. Capuano, et al., "ABITS: An Agent Based Intelligent Tutoring System for Distance Learning", *Proceedings of the International Workshop on Adaptive and Intelligent Web-based Educational Systems*, Montreal, Canada, 2000.

[7] W. Hodgins. Draft Standard for Learning Object Metadata. [S.l.], 2001.

[8] Russell, S.; Norvig, P. Artificial Intelligence: A Modern Approach. Prentice Hall Series in Artificial Intelligence, New Jersey, 1995.

[9] E. L. Ragnemalm, "Student Diagnosis in Practice; Bridging a Gap", Department of Computer and Information Science, Linkoping University, Linkoping, Sweden, Technical Report 1995.

[10] L. Fausset. Fundamentals of Neural Networks. Prentice Hall, Upper Saddle River, New Jersey, 1994.

[11] F. Bellifemine, G. Caire, T. Trucco, G. Rimassa, "JADE PROGRAMMER'S GUIDE", http://www.iro.umontreal.ca/~pift6802/Jade/doc/programmersguide.pdf, 2003. (Accessed in March, 2003).

[12] FIPA, "Foundation for Intelligent Physical Agents", http://www.fipa.org/. (Accessed in July 2003).

[13] G. Weiss, "Multiagent Systems – A Modern Approach to Distributed Artificial Intelligence", MIT Press, 2001.

[14] Marzo, J.L.; Peña, C.I.; Aguilar, M.; Palencia, X.; Alemany, A.; Vallès, M.; Johè, A. (2003) "Adaptive multiagent system for a Web-based tutoring environment." Relatório Técnico, Broadband Communications and Distributed Systems Group, University of Girona, Spain, 2003.Access: http://eia.udg.es/~atm/bcds/pdf/agentcities-final-report.pdf (July, 2003).