A Simulated Student Model for Improving Collaborative Learning

Aurora Vizcaíno Barceló
Escuela Superior de Informática
Paseo de la Universidad 4, 13071 Ciudad Real (Spain)
avizcain@inf-cr.uclm.es

Abstract

Learning in collaboration has numerous advantages. However, sometimes situations arise that may damage collaboration. This is the case for instance when there are passive students in a group. New technology may be used to detect and avoid some of the situations that hamper efficient learning in a group. This paper describes the model of a simulated student in charge of controlling and fostering students' learning in collaborative environments. An evaluation of the model was carried out. The results of the experiment indicated that in most cases the Simulated Student detected and corrected students' problems related to their learning.

Abstrait

L'apprentissage en collaboration a de nombreux avantages. Cependant, parfois les situations surgissent qui peuvent endommager la collaboration. C'est le cas par exemple quand il y a les étudiants passifs dans un groupe. La nouvelle technologie peut être employée pour détecter et éviter certaines situations qui entravent l'étude efficace dans un groupe. Cet article décrit le modèle d'un étudiant simulé responsable de contrôler et de stimuler l'apprentissage des étudiants dans les environnements de collaboration. Une évaluation du modèle a été effectuée. Les résultats de l'expérience ont indiqué que dans la plupart des cas l'étudiant simulé a détecté et a corrigé les problèmes des étudiants liés à leur étude.

Keywords: Cooperative systems and distributive environments, Educational system modelling.

1. Introduction

Collaborative learning has well-known benefits. However, when students learn in a group situations may arise that reduce collaboration between the students or impede appropriate learning. In order to control these situations this paper proposes a model for a simulated student which controls the students' behaviour and acts when a situation that debilitates collaboration or learning is detected.

The contents of this paper are organised as follows: in section two, the advantages of using simulated students in collaborative learning environments are explained. Section three describes the simulated student model that we have developed in order to foster collaboration and efficient learning. Section four outlines part of a profound evaluation of the model carried out using a real system. Finally a discussion about the results is presented.

2. Agents and Simulated Students in Intelligent Tutoring Systems and in Collaborative Learning Systems

The move from intelligent tutoring systems to pedagogical agents began about ten years ago, when researchers began to explore new types of interactions between computers and students (Johnson, 1999). Agents have played different roles, such as learning companions, teachers, advisors, etc. This work focuses on a type of agent that simulates student's behaviour and interacts with human students as if it were a real student.

Simulated students have only recently become available, even though the idea is not new (Doak and Keith, 1986). Simulated students improve traditional teaching methods where students work together in pairs or small groups (VanLehn, Ohlsson and Nason, 1994), the following paragraphs explain different advantages of using them in tutoring systems.

- Teachers can practice the art of tutoring by teaching a simulated student.
- Students can learn in collaboration with a simulated student.
- With a simulated student as part of the group, all kinds of pedagogically beneficial interactions can be staged from within the group itself- thought provoking questions can be asked, taciturn students can be prodded to speak, bad ideas can be questioned, small slips can be caught before they have serious consequences, attention can be directed away from areas that are already mastered and towards areas where students are ripe to learn (VanhLenh, Ohlsson and Nason 1994).
- A simulated student usually has one thing that a real student can never have: an expert knowledge database related to the problem to be solved. The lack of such expertise in a group composed only of human students dooms it to be less effective than one with a simulated student, in principle at least (Webb, 1989).

A simulated student can also monitor group interventions, detect miscommunications, and correct misunderstandings. Another very important advantage of using simulated students in collaborative environments is that each group may have its own simulated student (it is very difficult to have a human teacher monitoring each group because normally schools do not have enough teachers to do this). Besides, this simulated student is available at any time, so students do not need to worry about whether their partner or the teacher is busy.

3. A Simulated Student which Fosters Collaboration and Learning

This section describes a model for a Simulated Student which detects and avoids situations that hamper collaboration or learning in a CSCL environment. The Simulated Student controls the students' actions, analyses them and checks students' knowledge in order to encourage students to participate, or to help them to solve the exercises when they have problems in finding the solution.

The desired behaviour of the Simulated Student influences the configuration of the model. In the model outlined, the Simulated Student has a similar status to the human students (see Figure 1). This is an important feature, which creates several advantages such as favouring a more comfortable environment for collaboration or encouraging the students to reflect. Students feel more at ease working with friends who have a similar level of knowledge to themselves. Goodman et al., (1998) carried out an experiment which showed that the interaction between students is greater than interaction with a teacher. When students receive advice from a peer, students usually reflect on and consider the proposal but if the proposal is from a teacher, students do not generally query it.

The model described in Figure 1 shows the Simulated Student and the human students using the same methods of communication. This makes it different from previous models where the students perceived the agents as an animated figure which used another window to communicate with the students (Johnson et al., 1998; Goodman et al., 1998). To use the same techniques of communication for both real and Simulated Student helps students to consider the Simulated Student as a partner. In the previous cases, students could see the Simulated Student as an "assistant" offered by the system.

3.1 Description of the Model

The Simulated Student model has three main components: the individual Student Model (SM), the Group Model (GM) and the Simulated Student Behaviour

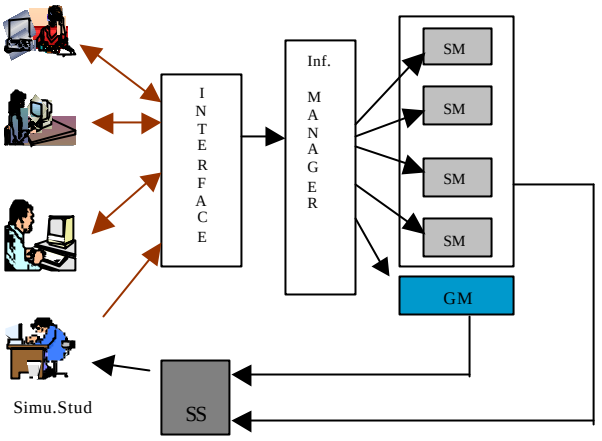

Figure 1. Overview of the model showing flows of information

Model (SSBM). It also has two complementary modules: the Information Manager and the Interface. The Interface is the means of communication between the (real and simulated) students. The Information Manager module classifies the information and stores it in the student models and group model. The SSBM uses the information stored in the Group Model and in the Student Models to decide when and how the Simulated Student has to intervene. All the three main components are described in detail in the next few paragraphs.

3.1.1 The Student Model

There is a huge amount of literature on student modelling. VanLehn (1988) defined the student model as the component of an Intelligent Tutoring System (ITS) that represents the current state of a student. Another definition is that proposed by Self (1994) which states that the student model is an element that tries to give information about the students.

A generic student model is formed from a set of entities where each entity expresses information about the learner. The information that the student model contains depends on the goals of the ITS. In our case, the goals are to detect and correct situations that decrease motivation and communication in collaborative learning.

If the student model is going to work in a collaborative application, further entities must be considered such as individual student's goals or opinions about their partners (Paiva, 1997). The student model that we have designed contains new entities that allow a greater control of the students' behaviour. The entities utilised are:

Frequency of interaction: One challenge of groupware applications is to provide a collective and equitable benefit. As Grudin (1994) claims, there is often a disparity between who does the work and who gets the benefit from it. Equitable and regular participation increases the amount of information available to the group, enhancing group decision making. Improving each student's frequency of interaction increases the likelihood that all group members will learn the subject matter, and decreases the likelihood that only a few

students will understand the material, leaving the others behind (Soller et al., 1998). Because of this, controlling the frequency with which a student interacts with the system and with his/her group mates is very important. The frequency of interaction is a critical factor for detecting passive students

Type of interaction: Student interaction may be of different types such as talking via a chat window (this may also be of different types, e.g.: proposal, question or an explanation) or solving exercises in the shared window. Knowing the type of interaction helps to characterise the student's role.

Level of knowledge: The knowledge that a student has is a factor that a ITS should take into account since it would have to adapt its exercises, explanations and, in general, the processing of learning to the student's knowledge.

Personal beliefs: In collaborative situations the learner's beliefs are not only about the domain but also about the other learners. One student's belief about another can produce an increase or decrease in the Zone of Proximal Development (ZPD) (Luckin and du Boulay, 1999). If a student thinks that his/her partner has more knowledge about a topic than him/her, s/he expects to learn more by working with this person than studying alone. Gracile (Ayala and Yano, 1995) is a system that uses mediating agents to exchange information about the students' skills and knowledge, trying to maximise the ZPD.

Mistakes: The detection of individual mistakes is very important in determining individual misconceptions. If one student makes less mistakes at the end of the session than s/he did at the beginning, this might indicate that learning has taken place.

3.1.2 Group Model

The group model is defined as a way of capturing those aspects that identify the group as a whole (Paiva, 1997). Different opinions exist about how to model the group. Paiva (1997) claims the group is something more than the sum of its parts. For her the group model must be constructed using the actions and beliefs with which the group is in agreement as a basis. A problem with this is how to initialise the group model when the session is beginning. Hoppe (1995) proposes an alternative using individual results to parameterise the group situation.

The group model used is based on Paiva's proposal because the amount of information stored in the entities proposed by her model is sufficient for the Simulated Student to be able to detect negative situations and act efficiently. The entities are:

- Group knowledge: Beliefs that the group has. These are inferred from the group's actions.
- Group mistakes: The mistakes diagnosed from the group actions are group mistakes. "However, it may be the case that misconceptions that are ascribed to the group are not shared by all the individuals of the group, since the group beliefs are the "accepted" beliefs, and thus may not be held by all" (Paiva, 1997, page 218).
- Differences: The differences between the students are an important factor to consider. An example of difference is the conflict where one student supports theory 'P' and another learner believes the theory 'not P'. On many occasions it is convenient to use the differences between students to trigger possible discussions. This strategy is used by COLER (Constantino-Gonzalez and Suthers, 2000).
- Preferences: To know what type of exercises or what kind of assistance students prefer permits the application to be adapted to its users.

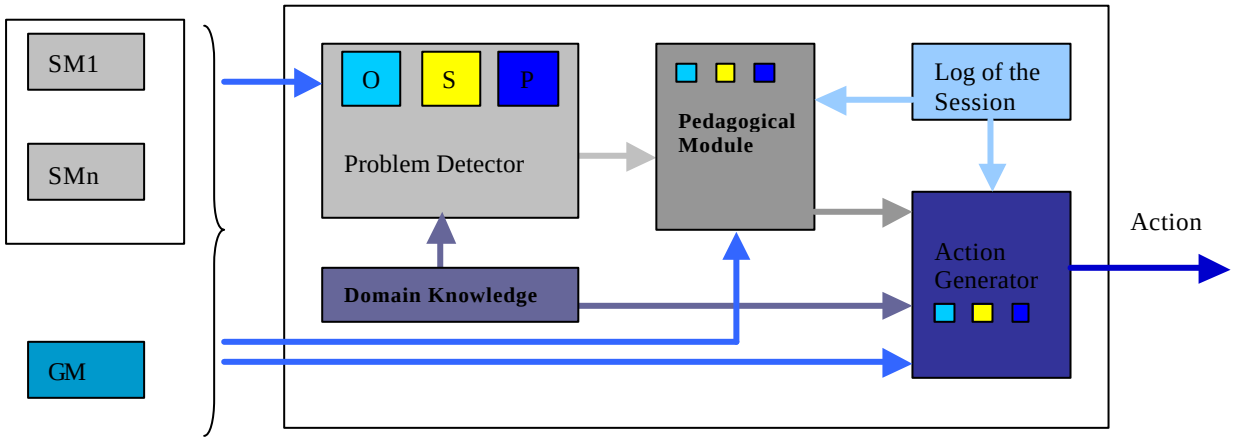

Figure 2. SSBM architecture showing flows of information

3.1.3 The SSBM.

The Simulated Student Behaviour Model (SSBM) is the most important component in the model. This component uses the information from the student models and the group model to decide how it must act. The architecture of the SSBM is displayed in Figure 2.

The SSBM is formed of five components, the problem detector, the pedagogical module, the log of the session, the domain knowledge and the action generator.

Domain Knowledge, as its name indicates, contains information about the subject to be learnt. This information is necessary in order to know how much knowledge students must have at each moment and also to adapt the Simulated Student's actions.

The Problem Detector, through the information received from the student models, the group model and the knowledge domain, checks whether negative situations are taking place. This module, as Figure 2 shows, is formed of three sub-components. The first one is in charge of control if off-topic conversations arise. The second component monitors the group's learning process, (this module will be explained in the following section). The last one checks each student's participation in order to detect passive students. It is possible to modify these components or add new ones to avoid other different negative situations. This is one advantage of this model, thanks to its modularity.

The Pedagogical Module indicates what action the Simulated Student should carry out in order to avoid the problematic situation that the problem detector identified. Several factors are taken into account by the Pedagogical Module before it chooses an action. They are the nature of the problem detected, the individual and group features, and the Log of the Session. The Log of the Session stores all the interventions, including those of the Simulated Student. Having a record of interventions enables the system to know the Simulated Student's previous answers and therefore makes it possible not repeat answers or actions. The Log of the Session is also useful for analysing the (simulated or real) students' behaviour.

Examples of types of action that the pedagogical system can trigger are to motivate the students to solve the exercises or to reinforce students' learning. The actions are classified into three groups, one group per negative situation; this is represented by the squares in different colours inside the Pedagogical Module.

For each type of action that the Pedagogical Module chooses, there exists a set of possible roles that the Simulated Student can play. Depending on the student models, the group model, the Domain Knowledge, and the Log of Session, the Action Generator chooses which role the Simulated Student should play. For example, if the Problem Detector detects a passive student, the Pedagogical Module can advise that the human student be invited to collaborate. The Action Generator decides how the Simulated Student should invite the student, perhaps with a direct invitation or with a question, etc.

3.1.4 The Learning Problem Detector.

The "learning problem detector" monitors the students' progress to decide when the Simulated Student should intervene. For instance, if the students propose correct solutions, the Simulated Student can ask about the solution in order to check whether the students really understand the solution or if they have just arrived at the correct by chance. The group and individual knowledge indicates what topics the students understand individually and at a global level. Both the individual mistakes and the group's mistakes indicate which subjects the students do not comprehend. The preferences of the group are another parameter to be taken into account, since a group might always fail in the same kind of exercise because they do not know how to approach it appropriately, even though they understand the topic which the exercise is asking about.

The Learning Controller sub-module checks whether the students have problems with the topic in hand or whether they have reached an appropriate level of knowledge (the domain rules indicates what degree of knowledge students should have at each moment). When irregularities are found in the learning process, the Irregularity Selector investigates what is causing the anomaly. This information is passed to the Pedagogical Module which decides what pedagogical support the Simulated Student should offer.

The model also has two more modules: the off-topic conversations detector and the passive behaviour detector. They are not deal with in this paper because of space.

3.1.5 Roles of the Simulated Student to help students to learn

On some occasions, groups waste a lot of time trying to solve a problem in an incorrect way. The fact that students try different ways is a good pedagogical technique because students learn from their experiences, and a central part of the learning process occurs when students attempt to apply instructional material to solve problems for themselves (Anzai and Simon, 1979; Anderson, 1983). Important learning progress may occur when students encounter obstacles, work around them, and explain to themselves what worked and what did not (Anzai and Simon, 1979; Ohlsson and Rees, 1991). However, this type of learning has potential cognitive and motivational pitfalls. Students trying to solve problems sometimes expend much time and effort pursuing blind alleys because of errors or poor strategies. Of course, in some cases students may learn something valuable while searching for a solution. In many cases, however, such episodes leave students confused and frustrated. So if a group does not obtain feedback after

spending a lot of time working on a task, members may lose motivation and even abandon the activity, or begin to talk about other topics causing some group members to feel that they are wasting their time. The Simulated Student might avoid these negative effects by monitoring the students' knowledge and their learning process. When the Simulated Student detects that learners are not close to finding a solution it could give clues or explanations and even, if it is necessary, indicate the correct answer.

The presence of a Simulated Student in collaborative applications could also avoid the Group Think Effect, which is another negative situation that arises in collaborative environments. The Group Think Effect is produced when the group accepts an idea for social reasons or because it is easier to do so. If a Simulated Student asks why they accept a proposal or proposes wrong ideas with the goal of producing doubt, the Group Think Effect should decrease. Table 1 summarises situations that can take place in a collaborative learning process. The role of the Simulated Student and the pedagogic strategy used to control the problem are also shown.

Situation	**Type of Intervention**	**Example**
Students cannot find the solution.	Giving hints. Asking for feedback. Consulting system's help.	The index of an array starts with 0, doesn't it? Do you remember if the index of an array starts with 0? Why don't we have a look at the counterexample.
Even with the clues students cannot find the solution.	Giving solution with explanation.	The solution is j=0 because the index of an array starts with 0.
Students have different points of view.	Helping to analyse alternatives. Expressing Disagreement.	Perhaps, the mistake is in the index of the array instead of in the numbers contained in the array. I don't agree with Tom's proposal, in this case the first sentence is not printed.
Students find the solution at the first attempt.	Congratulations. Checking students' knowledge.	We are the best!!! Why was the solution j=0?

Table 1. SS Interventions to help students learning

Although the off-topic conversation detector and the passive behaviour detector have not been explained above, we are going to describe schematically how the simulated student acts when it detects some of these situations, so that the reader may have a global idea about the roles of the simulated student.

Table 2 summarizes the role of the Simulated Student when it detects passive behaviour. Before acting, the Simulated Student investigates why the passive student is not taking part in solving the exercises.

Situation	**Type of Intervention**	**Example**
Student with deficient knowledge.	Asking for feedback. Asking for justification. Checking knowledge.	Tom, didn't you understand the previous exercise? You seem confused. Peter, Do you mind explaining your solution to us, Tom and I don't understand it. Tom, this time you propose a solution in the answer window, ok?
Student with adequate knowledge.	Invitation to participate. Asking for feedback. Asking for explanations. Asking about preferences.	• Ann, you are very quiet. What do you propose? • What do you think about my proposal, Ann? • Ann, you aren't joining in much. Are you tired? • Don't you like this kind of exercise?

Table 2. SS Interventions to avoid passive behaviour

When students have off-topic conversations the Simulated Student has a very clear role to get students' attention back to the exercises.

Situation	**Type of Intervention**	**Example**
Students talk about other topics for a long time.	Closing the conversation. Giving a clue. Proposing a solution. Encouraging students to continue.	I don't like football. Let's finish this exercise. I think we have to write "new" in the solution, don't we? The solution is 13, lets try it. Let's try to solve all the exercises!!!!, ok?

Table 3. SS Intervention to avoid off-topic conversations

4. Evaluation

In order to evaluate the Simulated Student's efficiency at detecting and controlling problematic situations, a Simulated Student was implemented following the above mentioned model. The agent was introduced into HabiPro (see Vizcaíno et al., 2000; Vizcaíno, 2001) a collaborative distributed system to develop good programming habits. This section describes the experiment designed to test whether the behaviour of the Simulated Student was adequate, and whether or not it improved the learning process.

4.1 Objective

The main goal was to observe how the Simulated Student reacted when faced with certain negative situations and how the behaviour of the Simulated Student affected the other students' learning. Another objective was to evaluate the students' assessment of the Simulated Student's interventions. Given these needs, the aims of the experiment described here were to explore:

1.- The efficiency of the Simulated Student in detecting problems in the learning process and its efficacy in solving these problems.

2.- The effect of the Simulated Student on the students' learning.

3.- The Students' assessment of the Simulated Student.

This paper focus mainly on the results obtained when the first aspect was analysed.

4.2 Design of the Experiment

Students had to solve problems using HabiPro in two sessions. In the first session one group of students used a version of HabiPro with the Simulated Student and another group of students used a version without the Simulated Student. In the second session the students used the version of HabiPro that they had not used in the first session. The experiment is a within-subjects design which means that comparisons are made between two or more results obtained from different circumstances, but always from the same group, thus avoiding the characteristics of the subjects affecting the results.

4.3 Subjects

Forty-four students enrolled on the first course of the subject “Introduction to Programming”, in the first year of the Computers Science degree in Ciudad Real (Spain), took part in the experiment. Students were randomly divided into the two sub-groups, one subgroup started the experiment working with the version of HabiPro containing the Simulated Student and the other subgroup with the version without Simulated Student. The sub-groups were also randomly divided into couples. So we had two subgroups of eleven pairs.

4.4 Procedure

Each couple taking part in the experiment attended two sessions about one week apart. The sessions lasted approximately one hour. Each pair had to solve programming problems using a different version of HabiPro in each session. So, the eleven couples that used the version without the Simulated Student in the first session used the version with the Simulated Student in the second session, and vice versa. Each student worked from a computer and they communicated with each other using the chat window.

4.5 Did The Simulated Student Detect when Students Needed Help to Solve the Exercises?

One role of the Simulated Student is to help the students to solve the exercises when the learners do not have a high enough level of knowledge or they are lost. When this happens the Simulated Student gives clues, hints or proposes solutions close to the real one. In this section the degree of success of the Simulated Student in playing the role of adviser is analysed. Before analysing the results obtained when students used the version with the Simulated Student we are going to analyse with what frequency students needed help to solve the problems when they worked without the Simulated Student.

Data showed that in 59 situations students did not solve the exercises at the first two attempts. These results will be commented on later.

Table 4 shows the results obtained when students used the version with the Simulated student. The logs stored when students worked with this version were analysed in order to answer the following questions:

- How many times did the Simulated Student detect that students needed assistance to solve the exercises?
- Did the Simulated Student's intervention help students to solve the exercises?
- Did students always consider the Simulated Student's advice?
- How many interventions by the Simulated Student were necessary to solve the problem?
- Did the Simulated Student act when it was inappropriate to do so?

The first column in Table 4 indicates each pair's number. The second column, called "number of times that students needed help", indicates how many times a pair had "problems" in solving the exercise. By having problems, we mean that the couple proposed two wrong solutions. This information was obtained from the stored logs. These contained all the answers written in the answer window (even the incorrect ones), all the conversation in the chat window and all the Simulated

Student's interventions. The third column indicates how many times the Simulated Student detected the situation. The fourth column called "students solved the problem" shows how many times the Simulated Student's intervention seemed to help the students to solve the exercise. The fifth column indicates how many times the students ignored the Simulated Student's proposal. The sixth column indicates how many times the Simulated Student intervened in order to help students. The last column indicates how many times the Simulated Student acted unnecessarily, in other words, when the Simulated Student considered that its help was necessary although it was not.

By comparing the data obtained we could see that students had more problems in solving the exercises when they used the version with the Simulated Student than in the other case. This fact might be because the exercises were more difficult and longer in the version with the Simulated Student.

Couple	Number of times that students needed help	Detected	Students solved the problem	Students ignored the help offered	SS. interventions	SS intervened unnecessarily
1	4	4	4	0	4	0
2	3	3	2	1	3	0
3	1	1	1	0	1	0
	1	1	1	0	1	0
5	2	2	1	1	2	0
6	2	2	2	0	2	0
7	4	4	3	1	4	0
8	3	3	3	0	4	1
9	0	0	0	0	0	0
10	2	2	2	0	3	1
11	5	5	4	1	5	0
12	2	2	2	0	2	0
13	4	4	4	0	4	0
14	2	2	2	0	2	0
15	2	2	2	0	0	0
16	5	5	5	0	6	1
17	5	5	5	0	5	0
18	4	4	4	0	4	0
19	2	2	2	0	2	0
20	3	3	3	0	4	1
21	4	4	4	0	4	0
22	5	5	5	0	5	0
N	22	22	22	22	22	22
Sum	65	**65**	**61**	**4**	**69**	**4**

Table 4. Number of Times that Students Needed Help and SS's Interventions

The logs of the version without the Simulated Student showed that in 68% of the cases students found the solution at the third or fourth attempt, and most times they needed to consult the help offered by the system. However, in the rest of the cases the students, instead of reflecting upon the problem, started to talk about other topics. This might be the reason why the students solved less exercises even though the problems were easier than in the version with the Simulated Student.

Now the data obtained in the case that the students used the Simulated Student version are analysed. The results show that the Simulated Student always intervened when it was necessary (100% successful), the logs indicated that when students proposed a wrong solution the Simulated Student acted by suggesting a solution or asking a question related to the solution. The intervention of the Simulated Student helped students to solve the problem in 93.8% of the cases, 61 times out of 65. However students ignored the Simulated Student's advice 6.15% of the time, hence in these cases the Simulated Student's interventions was not efficient.

From Table 4 it is possible to deduce that one intervention from the Simulated Student was enough to help the students to solve the problem. Table 4 shows more interventions (69) because of the 4 times that the Simulated Student intervened unnecessarily. In the following section the possible reasons why the Simulated Student acted when it was not necessary will be analysed.

The data obtained from the experiment support that the Simulated Student helps students to solve problems, because although students did not know how to attack the problem in many situations, the Simulated Student's interventions helped students to find the solution. In fact, as Table 5 shows, students solved more exercises correctly than when they used the version without the Simulated Student. So, we can say that the model of the Simulated Student is efficient to detect and correct learning problems.

Mann-Whitney Test of both Session

SESSION	N	Mean Rank	Sum of Ranks
Without Simulated Student	22	18.05	397.00
With Simulated Student	22	26.95	593.00
Total	44		

Table5. Mean Rank of exercise results with and without Simulated Student

5 Discussion and Future Work

From the experiment it was observed that the Simulated Student always intervened when students could not solve the exercises. An example of the Simulated Student's intervention is shown in the following conversation. In this case the Simulated Student is proposing a solution. It does not impose its idea, leaving the students free to check the proposal or to ignore it. The Simulated Student is Student 3

Student1: I don't know how the "loop for" works.
Student2: Yes, I see that, we have tried a lot of possible solutions and none of them are correct.
Student3: I think that the index of the array must be 0. Let's try j=0.

Student1: Yes!!! Now I remember that the index of an array starts with 0 in Java.

On the other hand, although the Simulated Student intervened when it was necessary, it also acted four times when it was unnecessary. When and why did this occur? The Simulated Student has no natural language capability hence it cannot understand the chat conversation. The Simulated Student uses the information from the answer windows and the number of times that students check a solution in order to decide when to act. So the Simulated Student may propose a solution that the other student has just written in the chat window. This would be an unnecessary intervention and this is what in fact happened on the four occasions that the agent intervened inadequately. Students might think that Student3's behaviour was strange because it proposed the same idea that had already been mentioned in another way. However, in most cases students thought that Student3 wrote the sentence in the chat window at the same time as Student2, but Student2's intervention arrived earlier.

Everybody who has used a chat application connecting two or more people at the same time knows that such a chat conversation is not as logical as an oral conversation, since, except in the applications that use a turn talking protocol, chat users are not aware whether the others are writing at the same time, and neither of them knows in which order interventions will arrive. So, in this case the expectations of working in a chat helped us to mask a possible defect of the Simulated Student.

To prevent the Simulated Student from repeating something that has already been said we are exploring the use of techniques developed in the field of natural language processing. We have discussed this issue with CICESE, a Mexican research centre, which is in the process of analysing the conversations obtained from our experiments, in order to adapt a syntactic analyser for Spanish that thy have developed to help the Simulated Student achieve a better understanding of the conversations (Ibarra, Favela and López, 2000).

References

Anderson, J. R. (1983). *The architecture of cognition*. Cambridge, MA: Harvard University Press.

Anzai, Y., and Simon, H. A. (1979). The theory of learning by doing. *Psychological Review*, 86, 124-140.

Ayala, G., and Yano, Y. (1995) Interacting with a Mediator Agent in Collaborative Learning Environments. In Y. Anzai, K. Ogawa and H. Mori (Eds.) *In Symbiosis of Human and Artefact: Future Computing and Design for Human-Computer Interaction*. Advances in Human Factors/Ergonomic, Elsevier Science Publishers, pp: 895-900.

Constantino-González, M. A., and Suthers, D. D., 2000. A Coached Collaborative Learning Environment for Entity-Relationship Modeling. In G. Gauthier, C. Frasson and K. VanLehn (Eds.). *Proceedings of ITS 2000*. Springer, pp: 324-332.

Doak, E.D., and Keith, M. (1986). Simulation in teacher education: The knowledge base and the process. *Tennessee Education*, 16, 2, 14-17.

Goodman, B., Soller, A., Linton, F., and Gaimari, R. (1998). Encouraging Student Reflection and Articulation using a Learning Companion. *International Journal of Artificial Intelligence in Education*, 9 (3-4).

Grudin, J. (1994). Groupware and Social Dynamics: Eight Challenges for Developers. *Communications of the ACM* 37 No. 1, 92-105.

Hoppe, H.U. (1995) The Use of Multiple Student Modelling to Parameterize Group Learning. In J. Greer (Ed.) *Proceedings of AI-ED 95*. Washington, USA.

Ibarra, M.A., Favela, J. , and López, A. (2000) Syntatic-conceptual Analysis of Sentences in Spanish Using a Restricted Lexicon for Disambiguation. *MICAI2000: Advances in Artificial Intelligence*, Cairo O., Sucar, L., and Cantu, F. (Eds.) LNAI1793, Springer, pp 538-547.

Johnson, W.L., Richel, J., Stiles, R., and Munro, A. , 1998. Steve: Integrating pedagogical agents into virtual environments. SIGART Bulletin 8, pp. 16-21.

Johnson,W.L.,(1999) Pedagogical Agents. http://www.isi.edu/isc/carte/pedagogical_agents.html.

Luckin, R., and du Boulay, B. (1999). Ecolab: The Development and Evaluation of Vygotskian Design Framework. *International Journal of Artificial Intelligence and Education*, 10 (2), 198-220.

Ohlsson, S., and Rees, E. (1991). The function of conceptual understanding in the learning of arithmetic procedures. *Cognition and Instruction*, 8, 103-179.

Paiva, A., (1997). Learner Modelling for Collaborative Learning Environments. In du Boulay and Mizoguchi (Eds.), *Proceedings of AI-ED 97*. Kobe, Japan, IOS Press, 215-230.

Self, J. (1994).Formal Approaches for Student Modelling. In Greer, J and McCalla (Eds.) *Student Modelling: the Key to Individualised Knowledge-Based Instruction*. Springer-Verlag.

Soller, A., Goodman, B., Linton, F., Gaimari, R. (1998). Promoting Effective Peer Interaction in an Intelligent Collaborative Learning System. Proceedings of the 4th *International Conference on Intelligent Tutoring Systems* (ITS 98). San Antonio, Texas, pp 186-195.

VanLehn, K., Ohlsson, S., & Nason, R. (1994). Applications of Simulated Students: An Exploration. *Journal of Artificial Intelligence in Education*, 5(2), 135-175.

VanLehn, K. (1988) Student Modelling. In Polson, C., Richardson, J.J.,(Eds.) *Intelligent Tutoring Systems*, pp 55-77.

Vizcaíno, A., Contreras, J., Favela, J., and Prieto, M. (2000). An Adaptive, Collaborative Environment to Develop Good Habits in Programming. In Proceedings of the 5th *International Conference on Intelligent Tutoring Systems*. Montreal, Canada, pp 262-271.

Vizcaíno, A. (2001). Can a Simulated Student Avoid Negative Situations in Collaborative Environments?. In proceedings of the *First European Conference on Computer-Supported Collaborative Learning*. Dillenbourg, P., Eurelings, A. , Hakkaraine, K. (Eds.), Maastricht, the Nerthelands, marzo 2001, pp 704-705.

Webb, N. (1989). Peer Interaction and Learning in Small Groups. *International Journal of Educational Research*, 13, 21-40.