Implementation of Intelligent Agents with Mobility in Educational Robotics Settings

Ilkka Jormanainen [1], Chiara Moroni [2], Yuejun Zhang [3], Kinshuk [3], Erkki Sutinen [1]
[1] Department of Computer Science, University of Joensuu, Finland
[2] DIIGA (Dipartimento di Ingegneria Informatica, Gestionale e dell'Automazione), University of Ancona, Italy
[3] Department of Information Systems, Massey University, New Zealand
ilkka.jormanainen@cs.joensuu.fi

Abstract

Teachers working in robotics classes face a major problem: how to keep track on individual students' or even small groups' progress in a class of 30-40 students. A multi-agent environment to help teachers with this problem is based on having pedagogical agents to monitor students' interaction, robots' movements, and the construction and programming process of robots. Mobile interaction agents move in local area network delivering other agents' observations to teacher's visualization agent. Results of a linguistic analysis of an educational robotics setting show the possible problems in educational robotics settings where the proposed agent-based system could help.

1. Introduction

Educational robotics is successfully used for teaching in several school contexts. These small-scale computerized teaching tools have many advantages over the PC based tools which have traditionally been used to teach, for example, programming or engineering. Lego Mindstorms is one well-known example of educational robotics. This robot construction set is flexible and simple enough platform for building and programming even for novices. Lego Mindstorms offers facilities for wireless connection. The current version uses infrared communication, but the next version, Lego Mindstorms NXT, will have Bluetooth communication facilities, which make it ideal tool for mobile and ubiquitous learning environments.

Educational robotics sets let room for student's own creativity by emphasizing active learner as the center of the learning process. However, in a typical classroom setting, especially at the elementary level, a teacher might have 30-40 children to teach. When using educational robotics in large classroom settings, students are usually divided into groups of 3-4 students. A typical educational robotics project follows an iterative cycle of building, programming, testing, and evaluation. It is characteristics that groups proceed differently, being in different phases of the cycle at the same time. This causes difficulty for the teacher to notice the needs for intervention. Our approach is to use educational agents to help the teacher to focus his/her attention in potential problems. The problem can be generalized as follows [6]: *How could the robotics environment inform the teacher what students are doing and how they are progressing?*

In this article, we describe a prototype implementation of educational robotics environment which aims to support the teacher to focus his/her attention in potential problems in the classroom. The implementation is based on the concept described by Jormanainen et. al [6]. The prototype implementation contains two major tracks. First, we have implemented intelligent agents which inhabit in the IPPE programming environment and Lego robots' RCX units. Second, we have built a screen-based model of classroom setting with the tools provided by Empirical Modelling environment. The model works as a visualization agent in our system. Finally, we have implemented a mobile interaction agent, which has the ability to move from one computer to another, for example from a students' computer to the teacher's computer to report a learner problem observed by the other agents.

This article is organized as follows. Chapter 2 describes the basic concept of the environment as well as existing work in the field. Chapter 3 introduces tools we have used to realize the prototype of the system and discuss the actual implementation of the system. Chapter 4 presents an evaluation we have done to examine the feasibility of our solution. This chapter describes an observation of discursive interactions in an educational robotics setting, aimed to analyse what kind of problems the participants had to handle and where the proposed agent-based system could help. Chapter 5 concludes the paper and draws some directions for the future work.

2. Background and existing work

Jormanainen et. al [6] introduce an agent-based architecture for the educational robotics environment that helps the teacher to intervene into small groups' work, based on the observed student-to-student interaction, robots' movements, and the group's progress in constructing and programming their robots. Unlike most current pedagogical agents which are usually applied to computerized learning environments or simulated virtual realities, the agency technologies described in [6] and in this paper are used in a traditional classroom, a real physical world, in which students are working on a Lego project.

In this chapter, we examine the tools needed to implement the architecture. First, we will introduce the concept of educational robotics. Then, we will discuss about the pedagogical agents in this context. Finally, we will discuss related work in the field.

2.1 Educational robotics

In the past few decades, researchers and industries have developed a number of different educational robot kits designed to help learning in scientific fields such as mathematics, physics, engineering, and computer science. These kits typically contain all the components which are needed to construct an autonomous robot. Autonomous robots can communicate and move independently according to the program the user has constructed. A well-known example of these tools is the Lego Mindstorms robot kit. The Mindstorms kit includes an RCX unit (Robotic Command Explorer), an independent computer, which is the core of the kit. An advantage of the RCX unit is its flexibility. The unit can be programmed by using a variety of programming languages such as Java or Visual Basic. The uses of the Lego Mindstorms kit in different contexts have been widely reported. The educational principles that motivate the usage of educational robotics are rooted in Jean Piaget's theories of cognitive development [11]. Seymour Papert built on these theories in his notion of 'constructionist learning'. According to constructionist principles, the active learner is the center of the learning process. Learners can create concrete new knowledge and learn in the constructionist way by interacting with real world objects.

2.2 Agents and mobile agents

There is no agreement in the literature on a unified definition for the term "agent". People use this term slightly differently according to the context where agents are employed. For the robotics classroom setting described in this paper, we follow the definition by Giraffa and Viccari [4]. Agents exhibit some properties upon which a common consensus has been reached. These properties include [15]: autonomy, social ability, responsiveness, and proactiveness. In addition, characteristics such as adaptability, veracity, rationality, and mobility are also desired for agents.

Agents possessing mobility are called mobile agents. Mobile agents are usually software programs, which may be dispatched from one computer and transported to a remote computer for execution. The motivation for using mobile agents stems from a number of potential benefits, such as efficiency and reduction in network traffic, asynchronous autonomous interaction, interaction with real-time entities, local processing of data, and support for heterogeneous environments [8]. Software agents have long been applied in educational environments to provide learning support. Educational agents are a class of agents that assists a user in an education-related task. Agents can monitor progress, give instruction when needed, help organize students' work, and provide feedback for tutors.

2.3 Related work

The idea of using agency technology to monitor learning progress and providing learner help in a robotics classroom setting is not new. Previous work can be found in [3], where a multi-agent framework for distance support via the Internet in educational robotics is described. In this framework, the teacher can obtain information about the learners' work and can be informed automatically by the system when learners encounter difficulties. Advantages offered by the agency approach in educational robotics are summarized by George and Despres [3] as follows: the system and its functionalities can easily be distributed between tutor's workstation and learners' workstations, the modular aspect of multi-agent system ensures easy modification of an isolated agent's behavior and a multi-agent system provides an open architecture which allows for the integration of new agents if required. We follow this argument, believing that agency approach is the right choice for monitoring pedagogical activities in the robotics classroom. Based on the previous work done by other researchers, we further develop an agency system by not only focusing on physical manipulations of the robot, but also controlling learners' programming activities and using mobile agents for asynchronous interactions as well as Empirical Modeling for visualization of observations.

3. Implementation

The educational robotics classroom setting is a physical world, rather than a virtual world where usually software agents are situated. The physical locations and activities of all objects, especially interactions among group members, are complicated and hard to model. Automating all the agency processes is difficult, if not impossible. Currently we decide to focus on two student activities: programming and manipulating the robot. We can monitor the former by building an agent in the students' programming environment; we can also monitor the latter by embedding another agent into the robot's RCX unit.

The implementation of the system contains two major tracks: pedagogical agents to observe the learning environment and a visualization agent for the teacher to observe the processes in the classroom. The system can contain several agents which all report their observations to one visualization agent. A typical robotic environment contains several instances where agents can be embodied. Figure 1 presents the overall concept.

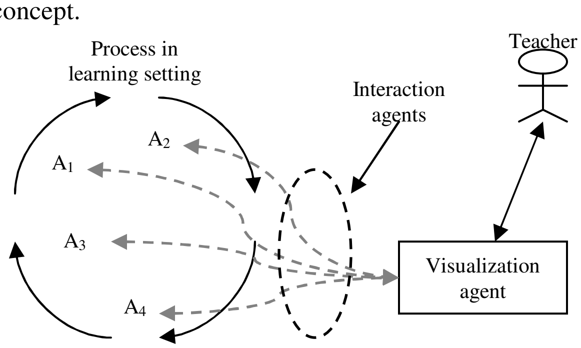

Figure 1. The overall concept of the system

In the Figure 1, the items $A_1 - A_4$ present the agents running in the learning setting and observing activities in it. The teacher can interact with the visualization agent to observe the state of the learning setting and control the behavior of the pedagogical agents according to the needs rising from the current learning situation.

Next in this section, we will present the IPPE programming environment and essential concepts of Empirical Modelling. These are the software tools we have selected as an implementation platform for our system. After that, we will describe the implementation of the agents more deeply.

3.1 The IPPE programming environment

One of the key elements with educational robotics is programming. The robots we have used, Lego Mindstorms, can be programmed with a variety of programming languages. For the implementation of our agent system, we have used the IPPE programming tool as a programming environment for robots. IPPE (Instructive Portable Programming Environment) is a tool for programming Lego Mindstorms robotics with a pseudo-like programming language near to the student's own natural language [5]. Programming can be done by composing the student's program with a graphical user interface, or by writing the code straight to the program editor window (Figure 2). The modular structure of the IPPE environment allows developers to write new features to support learning and teaching, such as intelligent agents described in this paper.

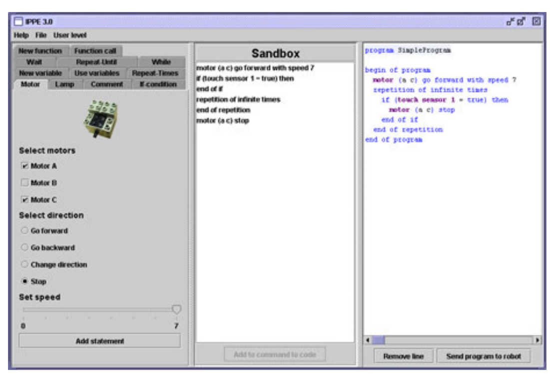

Figure 2. The IPPE environment

3.2 Empirical Modelling

An essential part of our system is a visualization agent for teacher's use. We use Empirical Modelling (EM) environment to implement this agent. The main reason to select EM as an implementation approach for teacher's agent rises from the nature of EM. The EM approach supports well a cyclic process and user's own observations about the phenomena. This means that teacher can update the model according his/her observations and, on the other hand, the teacher gets real-time feedback from the agents running in the environment.

The Empirical Modelling is an approach for constructing computer based models that can assist in the understanding of a phenomenon. The approach has an emphasis on experiment, observation and interaction during the development process [13]. Empirical Modelling describes the characteristics and features of a construal (model) with three key concepts: observables, dependencies and agents. This description is made by using a set of definitions, definitive scripts, and in this way, representing state-as-experience. The process of constructing computer-based artifacts using definitive scripts is called definitive programming. Interaction with EM models takes place through a continuously process of typing definitions on-the-fly. The modeling environment maintains automatically relations and dependencies between the different parts of the model. The approach has many similarities with a spreadsheet, where dependencies between cells are recorded by using definitions of the cells' content and their relations. The most used tool for building models with the EM approach is EDEN (Engine for DEfinitive Notations) interpreter called *tkeden*. The modeling process with core EDEN notation can be enhanced with different extensions. For example, *DoNaLD* and *Sasami* are definitive notations which allow usage of 2D and 3D graphics with EM models. In this work, we have used EDEN and DoNaLD notations to build the model of the classroom setting.

3.3 Implementation of agents

In general, the system described in this paper is a multi-agent architecture including four agent modules. The student agent inhabits the IPPE environment to monitor students' programming activities by reading data from the IPPE APIs on key strikes, mouse clicks, program errors and program types. The robot agent is seated in the Lego Mindstorms RCX unit to detect students' manipulations on the robot body by sensing signals from motors, sensors and buttons. The teacher agent makes decisions for the teacher based on the information reported from both the student agent and the robot agent, and implements visualization in which decisions made by the agent are presented to the teacher. The interaction agent works on the local network, facilitating the interaction among the student agent, the robot agent, and the teacher agent.

The interaction agent is designed to be mobile across the local network. It has the ability to move from one computer to another, for example from a students' computer to the teacher's computer, to report a learner problem observed by the student agent. The reason for the mobility of the interaction agent is to facilitate asynchronous autonomous interactions, so that the interaction agent can operate asynchronously and independently of the message sending agent (either the student agent, the robot agent, or the teacher agent).

3.3.1 Student agent. The student agent is relatively easy to implement. It collects data about student programming activities in the IPPE environment, while IPPE provides APIs to build software that handles events such as key strikes and mouse clicks. The main functionalities of the student agent include testing programming idle time, detecting programming errors and monitoring programming types.

The student agent model is straightforward. It takes student programming activities such as key strikes and mouse clicks as input, then outputs information according to the mission task and the input. The mission task described in certain specification language is received from the Empirical Modeling process. A new instance of the student agent will be created once a new mission is generated. One example of the mission description may be like:

```
delay-in-prog > 5
```

which indicates that the student agent will generate an alert whenever the programming idle time has over passed 5 minutes. The student agent model corresponding to this example mission task is illustrated in Figure 3.

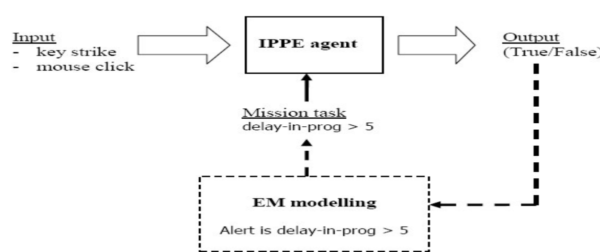

Figure 3. IPPE agent model

First, we implement a timer which signals at a fixed interval, in this case 5 minutes. Then we add a listener to this timer to handle the signal. In this way, the student agent will be able to give an output regularly. Finally, we implement a method which may reset the agent timer. To prevent the agent from creating an alert regularly, this reset method is triggered from the IPPE interface whenever a key is stroked or the mouse is clicked. The implementation of monitoring syntax errors in programming is also easy. For this purpose, the agent only needs to collect the debugging information from the IPPE environment via its APIs. However, monitoring semantic (or logic) programming errors and judging what type of programs the students are making is very difficult, due to the restrictions on current technologies.

3.3.2 Robot agent. The robot agent monitors students' physical manipulation of the robot by collecting data from motors, sensors, and buttons. Its main functionalities include monitoring the idle time for manipulation of the robot, detecting motor movements, and checking sensor signals. This agent resides in the robot control unit and sends data via an IR receiver to an interaction agent running on the student computer. Ultimately, the interaction agent delivers the agent data to the teacher agent and the Empirical Modeling process. The robot agent is implemented as a Java *thread*. The execution of the thread is started when the user starts running the program. The agent runs in the thread independently, gathering data from the RCX unit and building reactions based on it. The implementation of the agency is transparent. This means that the user does not notice the agent running in the robotics system. The agent is included in the user's program during the compilation process.

3.3.3 Teacher agent. The teacher agent in the teacher's computer presents agent data to the teacher. In this way, the system assists the teacher to achieve a better understanding about the progress of the students. The teacher agent contains two separate parts. First, the agent implements a visualization engine which maintains the classroom model built with the Empirical Modelling tools as described earlier in this paper. Second, due to the fact that EM tools have limited capabilities to communicate with the other applications, we will implement also a communication module as a part of teacher agent. This part will be implemented as a Java application in order to achieve a fully advantage with techniques such as RMI which takes care of the communication with interaction agents traveling in the local area network (see next section for more details). This communication module and the EM model will communicate through a local file system. The communication module passes agents' observations to the model as well teacher's definitions to the agents.

3.3.4 Interaction agent. The interaction agent is responsible for coordination among agents in general, and data transfer in particular. It is an independent Java program, which receives data from *the sender agent*, then moves towards the node *the receiver agent* resides and delivers the data locally to the receiver agent. When student activities are reported to the teacher, the sender agent is the student agent on a student computer, while the receiver agent is the teacher agent on the teacher computer. When a new mission task is generated and sent to a student computer, the situation is opposite. Figure 4 demonstrates the former scenario.

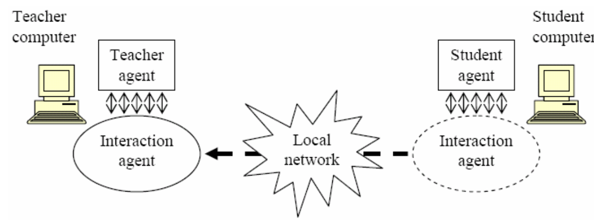

Figure 4. The mobile interaction agent

Each student computer as well as the teacher computer has an instance of the interaction agent running on it. We use Java RMI to implement the interaction agent. The Java RMI model allows an object running in one Java Virtual Machine (VM) to invoke methods on an object running in another Java VM. RMI provides for remote communication between programs written in the Java programming language. Java RMI is used as communication framework for mobile Java agents for several reasons [7]. Firstly, it supports multiple transport protocols like TCP and HTTP. Secondly, it provides a basic object registry / bootstrap mechanism. Finally, it moves code, not just data, across the network.

4. Evaluation of feasibility of agent system

To examine the feasibility of the proposed agent-based solution, we have carried out an observation in a real teaching setting where educational robotics was used. The aim of the observation was to examine the possible problems in educational robotics settings where the proposed agent-based system could help. The analysis was therefore finalized to detect the critical moments occurred during the sessions and to identify the strategies used by participants to handle them.

4.1 Setting

The observation was carried out during meetings of a technology club organised by Advanced Technology Research Centre at Massey University, New Zealand. The model for club meeting was taken from Kids' Club [2]. The club sessions were organized twice in a week during the school semester at the premises of Massey University. There were 10 participants in the club sessions, aged between 10 to 40 years. The participants were divided into three groups, and the groups solved a given technology-oriented tasks which contained planning, building, programming, and evaluating of a project with Lego Mindstorms robotics set. The participants came from diverse backgrounds having different amount of prior knowledge about building and programming.

4.2 Method and data collection

Discursive interactions are essential in every problem solving process occurring in social setting. In this section we propose a linguistic analysis of two club session interactions, in which shared problem solving efforts are observed from a pragmatic[1] perspective. The observation was conduct using linguistic and discourse analysis methodologies. In particular, pragmatic linguistic and conversational analysis[2] [10] offered interesting investigation tools to study discursive strategies used by participants in this kind of social setting interaction. The communicative interactions were observed as situated phenomena, with non evaluative but descriptive goals. We carried out a purely qualitative observation in a sample of transcription of videotaped club sessions.

Linguistic markers identifiable in the transcription text are the clues that let us know which *strategies* the participants have been using in order to manage the *problems* occurred during the sessions. Table 1 shows the observation scheme.

Table 1. Linguistics markers

Critical points	**Discursive strategies**	**Linguistic markers**
Possible problem occurring during the educational robotic setting	Strategies used by participants to handle the critical point	Signals in the text that unable us to recognize the strategies in each utterance

4.3 Analysis

Analyzing the transcriptions we recognized three principal kinds of problems, or turning points, that interlocutors had to manage during the observed sessions: lack of interaction, lack of comprehension and conflict between different opinions. Participants used mainly collaborative and negotiation strategies to deal with these critical moments.

4.3.1. Lack of interaction. We observed that the participants used strategies, such as conversation turn changing strategies, addressed to involve all the interlocutors into the discussion, in order to manage the lack of interaction occurring in some critical point of the meetings.

Table 2. Lack of interaction

Critical point	**Discursive strategies**	**Example of linguistic markers**
Lack of interaction	Strategies to enhance interaction and to involve all the participants	Discursive markers of conversational turns

In face-to-face conversations, there are three main strategies for change of speaker turn [14]: the current speaker may use names or vocatives, posture, or direct questioning to select the next speaker; the next speakers may select themselves; if no one self-selects, the current speaker may continue speaking. In the videotaped interactions we observed that the current speakers often used direct questioning to the other participants in order to give them the turn. As shown by the following sentences draw from the transcription, the current speaker can refer to the entire group:

M: Do you have any ideas?

or to a specific interlocutor, using vocative

C: What do you think Mc?

4.3.2. Lack of comprehension. "The process of mutual comprehension, is not a yes/no process, but involves intermediate stages, and evolves in the interactional development. Misunderstanding is a possible step in the process of understanding" [1].

In the observed sessions, to make the comprehension easier and avoid misunderstanding, participants used several discursive strategies.

Table 3. Lack of comprehension

Critical point	**Discursive strategies**	**Examples of linguistic markers**
Lack of comprehension	Strategy to avoid misunderstanding	- Repetition - Paraphrase - Shared encyclopaedia

They for example repeated their own sentences:

[1] Pragmatics is the research field focused on the study of language conceived as action in social context. From a pragmatic point of view "discourse" is "language in use".

[2] The central purpose of Conversational Analysis is to investigate the norms and conventions that speakers use in interaction in order to achieve communicative understandings.

A: maybe we should take it out. Take it out…Take this out.

Alternatively, they used paraphrase and referred to a shared lexical encyclopaedia:

O: We should make a little bit further maybe
A: Mm..?
O: Make this one going little bit longer.

4.3.3. Conflict. In the educational robotic setting observed, participants brought different ideas and opinions. The confront between different perspectives is never harmful; on the contrary, these conflicts of communication can improve the quality of groups interaction. Interlocutors advanced conceptually by reasoning and arguing collaboratively on different beliefs and ideas. [9]

To handle the confrontation between different opinions, participants used negotiation and argumentative strategies in order to challenge, claim their position, to express agreement or disagreement, and to compromise and integrate ideas.

Table 4. Conflict

Critical point	Discursive strategies	Examples of linguistic markers
Conflict	Argumentative and negotiation strategies	Discourse markers of: - Agreement/Disagreement - Argument in support of position - Asking/giving clarification - Asking/giving correction - Negotiation (compromise, rise-above, integration) - Summary

The contribution of individual knowledge and ideas was determinant to reach the shared solution. In some cases, it was just the opposition of points of view that contributed to create a part of the solution, as we can see in the following sentences, where the solution proposed by M came after a sequence of disagreement (discourse marker: "*but*"), and partial concession (discourse markers: *repetition* + *but*) between C and S:

C: eh… when… we need something that touch… when candy falls in…
S: but the candies are very light
S: when the red goes pass the light sensor… at the same time it will turn
C: at the same time but we don't know eh… for example if the candy is here (…)
M: well emm… first of all we should add the light sensor just before the fall, so that we could stop the thing… just... you know after it recognize… after it identifies the right one. It will move just a little that candy fall here and we stop this one.
S: oh! Until the first candy falls down
C: ok!

The role of contradiction and negotiation in problem solving setting is shown also in the following extracts. While in the first sequence the *repetition* of the own sentence and the discursive marker "*No*!" indicate disagreement between the two interlocutors, in the second sequence, the solution is finally reached through sentential cooperation (O completed the idea of A).

A: I was too fast
O: It was wrong
A: No, I was too fast, too fast maybe
O: No, it seems sometimes it do something wrong
(…)
A: I have to wait more then half second.
O: After it stops again.

The last extract shows therefore that participants reached solutions through forms of shared thinking [12]: knowledge was distributed across participants and their interactions determined the problem solution.

4.4 Results

The analysis allowed us to identify three main types of problems occurring in this kind of educational robotics setting: lack of interaction, lack of comprehension and conflict between different points of view. Furthermore, the observation shows how cooperation and negotiation strategies are essential to come through critical moments and to improve the problem solving process.

Moreover, the observation of the discursive interaction in these educational environments enabled us to detect the problems that the proposed educational agents could be able to support properly. The results of this study indicate that, beside the functionalities described in this paper, the agents should be also able to monitor the interaction amongst the group. In educational robotics settings, the interaction between students and groups' problem solving methods are crucial from learning point of view. Thus, agent system should be also able to monitor these activities. In particular, all kind of speech recognition technologies combined with the agent technology described in this paper could provide tools to automatically detect the problems in interaction amongst group members.

The agent implementation described in this paper does not answer to these needs. The proposed implementation has been meant to observe students' actions with the programming environment and robots.

However, the framework used to implement the system described in this paper, can be extended with such technology which allows monitoring for example dialogue between members of the groups. From that data, student agents could identify the problems in interaction amongst group members and build decisions suggesting collaborative and negotiation strategies.

5. Conclusions

Educational robotics provides ideal tools for mobile and ubiquitous learning environments. However, teachers have to adopt new kind of teaching methods and traditional classroom situations do not fit very well to educational robotics settings which emphasis group-oriented learning and problem-based methods. Teacher may face difficulties to notice the needs for intervention.

In this paper, we have presented implementation principles of agent-based educational robotics environment. The implementation is based on the description presented by Jormanainen et. al [6]. The implementation of the system contains two major tracks: pedagogical agents for observing the learning environment and a visualization agent for the teacher to observe the processes in the classroom. Proposed multi-agent architecture includes altogether four separate agent modules. We use the Lego Mindstorms educational robotics set including a small-scale computer (RCX unit) and the IPPE programming environment as host platforms for the agents. For implementation of agents, we use Empirical Modelling environment and Java programming language.

We have conducted a linguistic analysis to examine the possible problems in educational robotics settings where the proposed agent-based system could help. Based on the results, we argue that the physical locations and activities of all objects in learning environment, especially interactions among group members, are complicated and hard to model. Automating all the agency processes is difficult, if not impossible. However, proposed agent implementation has been meant to observe students' actions with the programming environment and robots and it can be extended with technology which allows monitoring for example dialogue between members of the groups. From that data, student agents could identify the problems in interaction amongst group members and build decisions in similar ways as described in this paper.

6. References

[1] Danieli, M., Bazzanella, C. (2002). Linguistic markers in coming to understanding, *Proceedings of VIII Meeting of AIIA(Associazione Italiana Intelligenza artificiale), AIIA 2002,* Siena, Italy, September 10-13.

[2] Eronen PJ., Sutinen E., Vesisenaho M., Virnes M. (2002). Kids' Club as an ICT-based learning laboratory. *Informatics in Education*, 1 (1), 61-72.

[3] George, S., Despres, C. (1999). A multi-agent system for distance support in educational robotics. *Proceedings of the International Conference on Telecommunication for Education and Training*, 344-353.

[4] Giraffa, L. M. M., Viccari, R. M. (1998). The use of agents techniques on intelligent tutoring systems. XVIII *International Conference of the Chilean Society of Computer Science*, 76-83.

[5] Jormanainen I., Kannusmäki O., Sutinen E. (2002). IPPE - How to visualize programming with robots. *Proceedings of the Second Program Visualization Workshop*, University of Aarhus, Department of Computer Science, DAIMI PB - 567, 69-73.

[6] Jormanainen, I., Zhang, Y., Sutinen, E., Kinshuk. (2006). Agency Architecture for Teacher Intervention in Robotics Classes. *Proceedings of the 6th IEEE International Conference on Advanced Learning Technology (ICALT 2006).* To appear.

[7] Kiniry, J., Zimmerman, D. (1997). A hands-on look at Java mobile agents. *IEEE Internet Computing, 1*(4), 21-30.

[8] Kinshuk, Hong, H., Patel, A. (2002). Adaptivity through the Use of Mobile Agents in Web-based Student Modelling. *International Journal of E-Learning*, 1(3), 55-64.

[9] Mason, L. (2001). Introducing talking and writing for conceptual change: a classroom study, *Learning and Instruction*, 2001, 11, pp. 305-329.

[10] Mazur, J. M. (2004). Conversational analysis for educational technologists: theoretical and methodological issues for researching the structure, processes, and meaning of on line talk in Jonassen, D.H. (ed) *Handbook of research on educational communications and technology*, Laurence Erlbaum Associates, Mahwah, New Jersey.

[11] Miglino, O., Lund, H. H. (1999). Robotics as an Educational Tool. *Journal of Interactive Learning Research*, 10(1), 25–47.

[12] Resnik, L.B., Salmon, M., Zeitz, C. M., Wathen, S. H., Holowchak, M. (1993). Reasoning in conversation. *Cognition and instruction*, 11(3/4), 347-364.

[13] Roe, C. (2003). Computers for Learning: An Empirical Modelling perspective. Doctoral dissertation, Department of Computer Science, University of Warwick, UK.

[14] Sacks, H., Schegloff, E., Jefferson, G. (1974). A simplest systematics for the organization of turn-taking for conversation, *Language*, 50(4), 696-735.

[15] Wooldridge, M. and Jennings, N. R. (1995). Intelligent agents: theory and practice. *The Knowledge Engineering Review*, 10(2), 115-152.