

A Multi-Agent Internet Based Tutoring System (I-ATCL) for Teaching Computer Programming Languages

Mahmoud M. El-Khouly, Behrouz H. Far and Zenya Koono

*T 338, Urawa-Shi, Shimo-Okubo, 255, Saitama-ken, Japan.
Saitama University, Faculty of Engineering, Information and
Computer Sciences Department.
{elkhoully, far, koono}@cit.ics.saitama-u.ac.jp*

This paper presents an agent-based tutoring system (I-ATCL) in presence of agent and internet technology. The I-ATCL system consists of two agents; a personal assistant agent for teachers (PAA-T) and a personal assistant agent for students (PAA-S). The PAA-S resides on the client side and communicates via IIOP with the PAA-T on the server side. This structure allows customization of the PAA-S to the needs of the student without putting extra burden on the server. Also this allows having many teacher agents attending to the needs of a single or multiple student agents. The PAA-T allows the teachers to cope with the knowledge base of a computer language under investigation, add or modify the commands' structure that will be taught. Also, this agent can generate a new tutoring dialog for a new computer programming language by consulting previous tutoring dialogs for another computer programming language. PAA-S contains student model and tutoring module. In student model, the system can accept free-format answer from the student, and checks it against the language structure.

Keywords: **CAI, ITS, Software Agent.**

1 Introduction

Learning one of the computer programming languages is essential for students in both undergraduate and graduate levels. However, the lack of personalized instructing material that match the knowledge of the students is a major problem. This problem can be removed by using computer-aided instruction (CAI) and Internet technology.

With the WWW as an educational platform, it will be feasible for the students to access the multimedia courseware with general-purpose browsers. No special tools are required to start learning. For the courseware provider, it is not necessary to worry about the distribution and maintenance of the copies of the courseware but they just take care of the original on their server [4].

Lewis Johnson [5] supports interaction with teachers and students through his project ADE, using off-the-shelf whiteboard and teleconferencing tools. But this required both teachers and students to be on-line.

The proposed system consists of two agents representing a server-client relationship, personal assistant agent for teachers (PAA-T) as a "server", and personal assistant agent for students (PAA-S) as a "client". First, a human expert produces the semantic rules (SR) for a computer programming language to be taught, then the PAA-T searches about previous tutoring text (PTT) for some/all of these semantic rules. If it finds some/all then it produces the tutoring text base (TTB), otherwise it asks the expert to

provide it. The PAA-S can communicate with PAA-T through WWW to retrieve the tutoring dialog of the command(s) which a student wants to practice. PAA-S consists of tutoring module (TM), student model (SM), and student's behavior-base (SBB) (as shown in Figure 1).

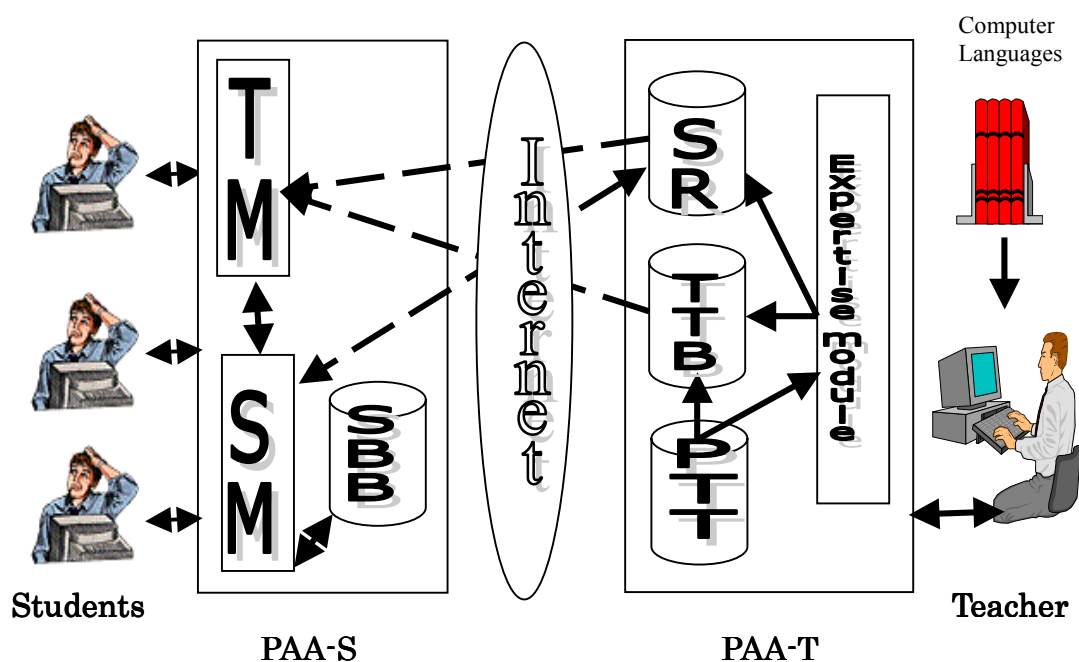


Figure (1): I-ATCL system

2 Personal assistant agent for teachers

The personal assistant agent for teachers (PAA-T) consists of four parts, expertise module, semantic rules base, tutoring text base, and previous tutoring text base [6]. PAA-T helps the teacher to cope with the knowledge base of a computer language under investigation, to add or modify the command's structure that will be taught, and to produce a meta level language representing this computer language, which we call it "semantic rules".

2.1 Semantic rules

When the teacher wants to build the semantic rule base, the screen, which describes building the semantic rules base will appear containing the keys guide to build the knowledge base.

In this phase, the teacher puts the structure of each command (using upper case characters for reserved keywords only), e.g.

IF vov THEN s

Which means that (*if* the condition (*variable operator variable*) is satisfied *then* do the statement *s*).

Tutoring module retrieves the structure of commands to be presented during the tutoring dialog. Moreover, the student model consults it to check the answers of the students.

3 Personal assistant agent for students

The personal assistant agent for students (PAA-S) consists of three parts, student model, student behavior base, and tutoring module.

3.1 Student model

The student model is used to assess the student's knowledge state and to make hypotheses about his/her conceptions and reasoning strategies employed to achieve the current knowledge state. Because most ICAI systems represent the student's knowledge state as a subset of an expert's knowledge base, the model is constructed by comparing the student's performance to the computer-based expert's behavior on the same task. This technique is named the "overlay model". Most early systems (e.g. WEST [1], SCHOLAR [2] and WUSOR [3]) used this approach.

3.2 Tutoring module

When the student accesses the module, the system asks about the computer's language, which the student wants to learn. The system opens the directory of this language and waits for the student to select the command s/he wants to practice, to retrieve the associated tutoring dialog's file(s). The system presents the text that describes the command, and asks the student if he understands it or not. If the reply is "no", the system converts to another style for presenting the command (e.g., Q&A). If the reply is "yes", the system asks the student to write an example statement to check it. The system can accept a free format text from the student, and then infers the student module to check this statement. If the statement is correct, the system increments two counters, one for the number of questions which had been asked to the student, and the other is the correct answer counter. Otherwise, only the first counter will be increased. If the correct answers of a student exceed 75% then the level of difficulty will increase by one, but if it is less than 50% the level of difficulty will decrease by one. After the student completes this step, the system re-presents the first menu to allow the student to select one of the followings: (a) select another command, (b) present the score, and (c) exit.

4 I-ATCL and WWW interface

In the stand-alone version [6], students receive immediate feedback on every action they take. If they enter an incorrect statement, the system will present a message error and guide them to correct answer. HTML forms do not allow this kind of tightly coupled interaction. PAA-T server receives information about student actions only when the student submits that information. In I-ATCL system, the PAA-T and PAA-S agents can find and communicate with each other dynamically, using Common Object Request Broker Architecture (CORBA) as depicted in Figure 2. CORBA allows agents find each other and coordinate their behavior on a common object bus. This makes CORBA ideal for component-based applications. The advantages of using such technology in I-ATCL system is using those objects as a metaphor for using existing stand-alone ATCL application as well as personalizing and task sharing between the client and server. In Internet based I-ATCL, the Java ORB is used. With a Java ORB, an applet can invoke methods on CORBA objects using the IIOP protocol over the Internet. Consequently, there is no need to use HTTP and CGI programs that are the cause of extra overhead on the server and also the client-side ORB enabled applets can be used in any Java enabled browser.

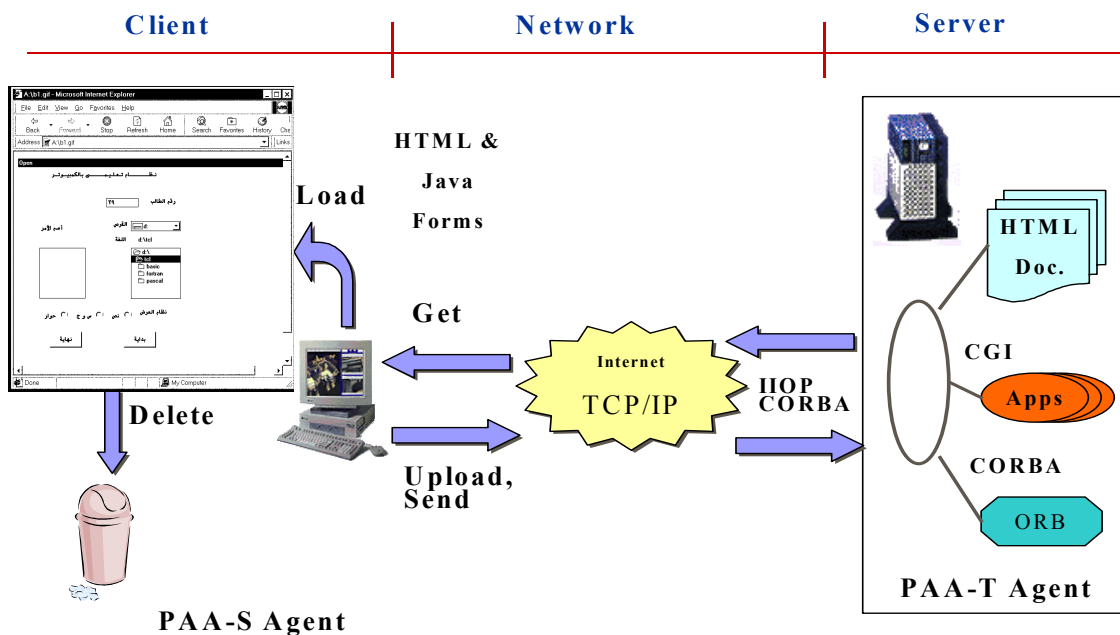


Figure (2): The I-ATCL system using ORB

5 Conclusion

In this paper, I-ATCL system had been presented for teaching computer programming languages. I-ATCL consists of two agents, personal assistant agent for teacher and personal assistant agent for students. They communicate with each other as client-server through WWW. This allows extending the features of the system to communicate with other agents to exchange semantic rules and tutoring text for different languages. In student model, the system can accept the free format answer from the student.

References

- [1] Brown J. and Burton R., "An investigation of computer coaching for informal learning activities", *Int. Journal of Man-Machine Studies*, 11,5-24, (1979).
- [2] Carbonell J.R., "AI in CAI: An artificial intelligence approach to computer-assisted instruction", *IEEE Transactions on Man-Machine Systems*, 11,190-202, (1970).
- [3] Goldstein I., "The genetic graph: A representation for the evolution of procedural Knowledge", In D. Sleeman and L. Brown (Eds.), *Intelligent Tutoring System*. London: Academic Press, (1982).
- [4] Kiyoshi Nakabayashi, Mina Maruyama, Yoshimasa Koike, Yasuhisa Kato, Hirofumi Touhei and Yoshimi Fukuhara, "Architecture of an Intelligent Tutoring System on the WWW", *Proceedings of the 8th World Conference of the AIED Society, Kobe, Japan, (1997)*.
- [5] Lewis Johnson and Erin Shaw, "Using Agents to Overcome Deficiencies in Web-Based Courseware", *AI-ED'97 workshop on intelligent educational systems on the world wide web, (1997)*.
- [6] Mahmoud El-Khouly, Behrouz H. Far and Zenya Koono, "Agent-Based Computer Tutorial System –An Experimental for Teaching Computer Languages (ATCL)-", special issue for *Intelligent Agents for Computer-Based Educational Systems of the Journal of Interactive Learning Research*, To be published.