PERGAMON

Computers & Education 40 (2003) 157–181

COMPUTERS & EDUCATION

www.elsevier.com/locate/compedu

Analysis and design of a Web-based authoring tool generating intelligent tutoring systems

Maria Moundridou*, Maria Virvou

Department of Informatics, University of Piraeus, 80, Karaoli and Dimitriou st., Piraeus 185 34, Greece

Received 9 April 2002; accepted 15 August 2002

Abstract

Authoring tools for intelligent tutoring systems (ITSs) are meant to provide environments where instructors may author their own ITSs in varying domains. In this way, painful constructions of ITSs, which are not reusable, may be avoided. However, the construction of an authoring tool is associated with many problems, such as the generality of the techniques incorporated, domain-independence, effectiveness for the prospective authors (instructors), and effectiveness for the students who will use the resulting ITSs. In this paper we will report on an empirical study that we conducted in order to design and develop WEAR, an ITS authoring tool for Algebra-related domains. In the study we investigated several aspects concerning the attitude and behaviour of both students and instructors. The study revealed important issues and was then used for the specification of the design of WEAR. A brief description of the developed system is also included in the paper so that the way that the design specifications were put into practice may be shown. However, a lot of the authoring tool's requirements that came to light could be applicable to other authoring tools as well. The most important requirement of this kind was the need for an instructor modelling component so that adaptivity could be provided to human instructors (authors). The provision of such facility is a novelty in the area of ITS authoring tools.

Keywords: Authoring tools and methods; Distance education and telelearning; Intelligent tutoring systems; Teaching/learning strategies

1. Introduction

Intelligent Tutoring Systems (ITSs) are computer-based instructional systems aiming at reproducing the behaviour of a human tutor who can adapt his/her teaching to the learning needs of

* Corresponding author.
E-mail addresses: mariam@unipi.gr (M. Moundridou), mvirvou@unipi.gr (M. Virvou).

0360-1315/03/$ - see front matter
PII: S0360-1315(02)00119-7

the individual student. As such, ITSs have the ability to present the teaching material in a flexible way and to provide learners with tailored instruction and feedback. This is achieved through the main components constituting an ITS: the domain expert, the student model, the teaching expert and the user interface. A number of successful evaluations of ITSs (Corbett & Anderson, 1992; Koedinger, Anderson, Hadley, & Mark 1997; Lajoie, 1993; Shute & Glaser, 1990) have shown that such systems can be educationally effective in comparison with traditional instructional methods either by reducing the amount of time it takes students to reach a particular level of achievement or by improving the achievement levels, given the same time on a task.

The main flaw of ITSs and possibly the reason for their limited use in workplaces and classrooms is the complex and time-consuming task of their construction. As estimated by Woolf and Cunningham (1987) an hour of instructional material requires more than 200 h of ITS development time. Furthermore, an already constructed ITS for a specific domain, can neither be reconstructed to function for a different domain, nor can it be altered (i.e. to reflect a different tutoring strategy in the same domain) without spending much time and effort.

An approach to simplifying the ITS construction is the development of ITS authoring tools. The goal of such tools is to provide an environment that can be used by a wider range of people to develop cost-effective ITSs. However, the high degree of difficulty and complexity of the construction of an ITS passes to the construction of an ITS authoring tool which is a yet harder and more complex task. The reason for this is that authoring tools should be able to operate effectively at two levels: at the first level, they should operate effectively for the instructors who intend to author an ITS and at the second level they should incorporate the expertise needed to produce ITSs that operate effectively for students. Moreover, they have to use domain-independent methods and be general enough to be used for the construction of many ITSs.

In view of the above, the development of an ITS authoring tool is an issue that needs a lot of attention. Knowledge elicitation and acquisition from domain expert tutors are needed for constructing the framework of the knowledge bases of the ITSs that will be generated by the authoring tool. However, it is widely acknowledged that the process of knowledge acquisition is a bottleneck in the development of expert systems (e.g. Boose, 1993; Garg-Janardan & Salvendy, 1988; Kitto & Boose, 1989; Walczak, 1998). In the case of ITSs, knowledge acquisition is even more difficult than in other expert systems because as Twidale (1992) points out, in ITSs knowledge acquisition is necessary for all three main elements: domain expert, teaching expert and student model. Finally, the case of ITS authoring tools increases the difficulty in knowledge acquisition since the expertise to be represented has to be as domain-independent as possible so that it may be used for ITSs of multiple domains.

Knowledge engineering techniques include protocol analysis, observations, interviews and introspection, case analysis and questionnaires (Bell & Hardiman, 1989). However, often the result is a ''data dump'' that leads to knowledge that is poorly structured, inflexible and incomplete (Thurman, Brann, & Mitchell, 1997). Therefore, in ITS authoring tools which are complex systems, reporting on the knowledge engineering techniques used, would be an asset for the advancement of this technology. However, reviewing the ITS literature one comes up with the finding that there is a serious lack of any reports on the development of ITS authoring tools. The lack of reports in the literature does not promote the future achievements in the area of authoring tools. Collins (1992) notices: ''We have had many technologies introduced in classrooms all over the world, but these innovations have provided remarkably little systematic knowledge or accumulated wisdom

to guide the development of future innovations''. By conducting empirical studies and presenting their results, future research efforts can both be advanced and made easier.

The rapid growth of the Internet and World Wide Web offers new opportunities and challenges for many areas. One of them is education. Web-based education has numerous advantages such as the convenience of taking a course without leaving the workplace or home and the reduced cost (Berz, Erdelyi & Hoefkens, 1999). In addition, teachers and educational researchers are encountering both unprecedented opportunities and challenges to adapt networks to their classrooms and research fields (Chou, 1999). However, most of the educational applications that have been delivered through the World Wide Web are just electronic books with very limited interactivity and diagnostic capability. An integration of ITSs and Web-based technologies would be very beneficial for the purposes of education. Indeed, there have been successful attempts to either move existing ITSs to the WWW (e.g. Alpert, Singley, & Fairweather, 1999; Ritter, 1997) or build from scratch Web-based ITSs (e.g. Eliot, Neiman, & Lamar, 1997; Peylo, Thelen, Rollinger, & Gust 2000). However, Web-based ITSs aim at reaching a much more heterogeneous group of learners in settings where no teacher is available to help users during the learning process. In order to match the large diversity of student needs, backgrounds and interests, a careful and extensive knowledge acquisition phase should precede the development of a Web-based ITS. This is even more urging in the case of Web-based ITS authoring tools which are also addressed to a wide range of instructors with varying beliefs, preferences and teaching styles. Indeed, as noted by Hendler and Feigenbaum (2001), the knowledge acquisition bottleneck becomes particularly troubling in the era of the World Wide Web and they continue by saying that the ubiquity of information accessible at the fingertips of millions of users drives a relentless demand for more intelligent computer assistance.

In this paper we will describe an empirical study that we conducted and led us to the development of WEAR, which is a Web-based ITS authoring tool. What we intended to produce was an authoring tool for building intelligent tutoring systems in algebra-related domains. As algebra-related domains we consider these domains that make extensive use of algebraic equations (e.g. physics, economics, chemistry, etc.). Before conducting the empirical study, we decided upon the basic requirements of the software that we would produce and we determined the aspects that needed further examination. These aspects formed the investigation study that we performed and will be discussed in this paper. In particular, we will describe the three parts which our study consisted of. The first part of the study aimed at identifying issues relating to the ITSs that the authoring tool would be generating and it involved mainly students. Students were given some problems to solve and the analysis of their answers by expert instructors gave rise to the design guidelines for ITSs in algebra-related domains. The second part of the study which involved only instructors aimed at identifying issues that related to the instructors' needs and expectations when authoring an ITS with an authoring tool. To achieve this, instructors were asked several questions concerning their preferences, teaching strategies and beliefs. The results of this second part of the study formed the design guidelines for WEAR's authoring environment. The third and final part of the study aimed at verifying the design guidelines of both the authoring and learning environments of WEAR. In this part both instructors and students were involved. A questionnaire was given to instructors who were asked to rate the functions of an ITS authoring tool that reflected the resulting from the first two parts of the study design guidelines. Students' involvement in this verification phase of the study, concerned the evaluation of the user interface

of the generated by WEAR ITSs. In the main body of this paper we will describe and discuss the whole study and we will present the operation of the developed system. Finally, we will give the conclusions drawn from this work.

2. Experiment concerning the construction of intelligent tutoring systems

2.1. Aims and settings of the experiment

The first part of the experiment aimed at identifying issues that related to the construction of a generic frame for ITSs. In particular, the main questions that had to be addressed concerned the domain knowledge, student models and the generation of advice of the ITSs that would be constructed automatically by the authoring tool. However, these issues had to be addressed in a way that would be as domain independent as possible. Therefore, one of the main targets of the experiment was to identify similarities in the problem solving approaches and categories of students' errors in these domains. The categories of error would serve as the basis for the design of the student modelling component of the ITSs. Finally, the experiment also aimed at identifying student characteristics that could be modelled and could be associated with the students' proneness to certain kinds of error and their general performance in problem solving. Such information would be useful for setting the aims of the inferential capabilities of the student modelling component.

The experiment involved high school students. At this point, it should be noted that in Greece three directions of graduate studies exist: the science, the technological and the arts direction. Students attend some classes that are common to all directions and some others that are specific for the direction that they have chosen. The experiment was based on courses that were common to all directions. In particular, two groups of 15–16 years old students were involved. The two groups originated from courses of two different algebra-related domains; one was from a physics course (35 subjects) and another one was from an economics course (20 subjects). In order for our sample to be unbiased most of the students who participated were from different high schools.

Students of each group were given problems from the corresponding subject and were asked to solve them. The students' answers to these problems were given to domain expert tutors to analyse them. The problems corresponded to what students were taught recently in their classes and were given to them as part of an informal exam. Students who participated in the experiment were also asked to provide their previous years' marks both in the corresponding subject (physics or economics) and in mathematics. This was done, so that the instructors who would analyse the students' protocols could be able to comment on the relation between students' past performance and the current performance.

Finally, students were asked to answer questions concerning the course that they were taught at their respective schools, their instructor and their interests. For example, they were asked what they were interested in knowing about the course and their instructor in the beginning of a course; whether they sought information from older students who had taken the same course before they did; whether they were interested in the course they were taking; what their intentions were concerning their studies after graduating from school. Students were requested to state the direction they were following because this kind of information would indicate the students'

degree of interest in the courses taught and would be useful for identifying possible relations of students' interests and their performance.

All this information as well as the papers that the students turned in, were analysed by human experts. There were four human experts who participated in this part of the experiment. Two of them had a first degree and an MSc in Physics and two of them had a first degree and an MSc in Economics. All of them had at least 4 years experience in teaching at schools. The four instructors were different from the instructors that the students had at their schools. This was done so that the students who participated would not feel intimidated or nervous.

2.2. Analysis of the results

Concerning the possible similarities in problem solving approaches of different algebra-related domains, the experiment revealed that there was a standard method for solving particular problems that were set to students of various levels:

- students should read and comprehend what is given and what is asked in the stated problem and they should form the equations(s) needed for the problem to be solved, and
- students should then solve the system of equations for the unknown variable(s) of the problem.

For example, in the domain of physics one problem of this kind is the following: "A force of 100 Newtons is acting on a 25 kg object, which is initially stable. After 10 seconds how much is the impulse?" For a student to solve this problem s/he should use the equations $F = m*a$, $J = m*v$ and $v = v_0 + a*t$. S/he should replace each known variable with its value and solve for the unknown variables. Similarly, in the domain of economics one such problem could be this: "Given that the Gross National Product–GNP is 1500, Net Factor Payments from abroad–NFP is 20, Investment–I is 250 and National Saving–S is 300, find the Gross Domestic Product–GDP and Net Exports–NX." Again, students should replace the given data in the equations $S = I + CA$, $CA = NX + NFP$ and $GDP = GNP - NFP$ and solve for the unknown variables.

In such kind of problem and irrespective of the specific domain to which these problems belong, each of the above-mentioned steps for solving the problem can be an error source. In particular, students may not form the equations correctly that are needed for the problem or they may not successfully solve the equations. The former kind of errors can be considered as "domain errors" and the latter as "mathematical errors". These two error categories can be further divided into sub-categories (Fig. 1). Concerning the "domain errors", the students' answers manifested three cases: either students may not be able to write down an equation at all, or students may attempt to use a correctly formed equation which is not appropriate for the particular problem, or students may not form the equation correctly. In this last case, the possible error is either that students may form an equation that involves one or more erroneous variables, or students may write down an equation where some variables are missing or not needed, or that students may confuse the relationship between the variables of an equation. Concerning the "mathematical errors", these are distinguished in calculation errors, replacement errors and errors in isolating the unknown variable of an equation.

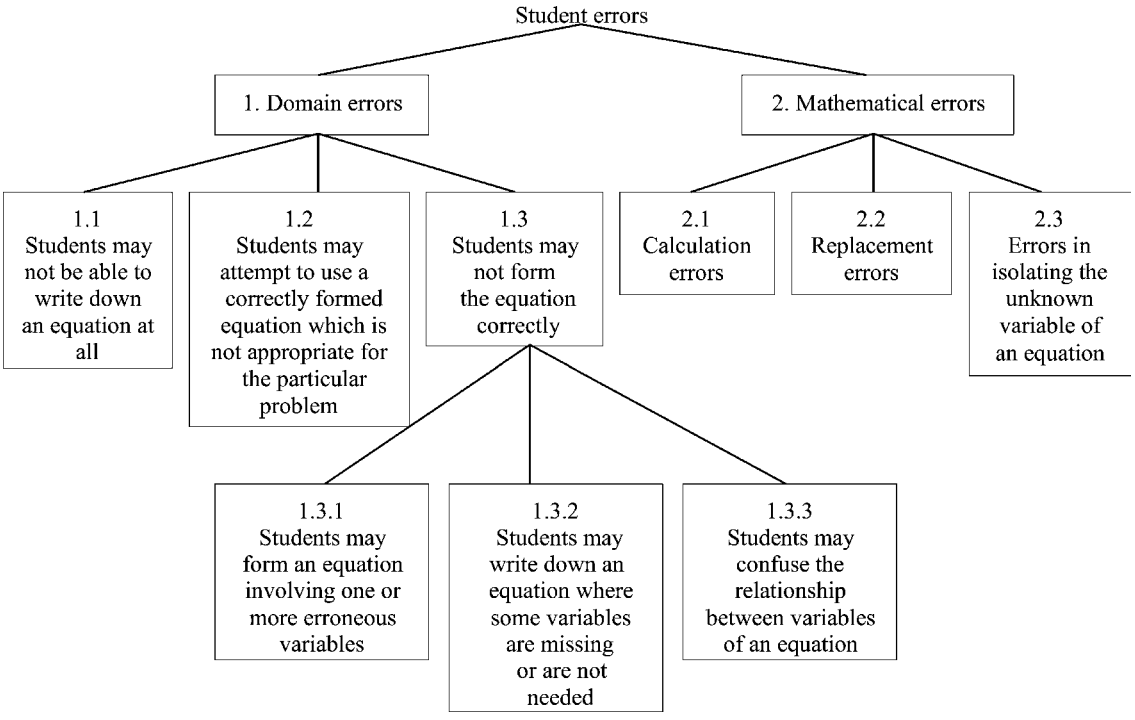

Fig. 1. Categories of student errors.

The experiment showed how the errors were distributed along the aforementioned categories and sub-categories of error in each of the domains of the study. These quantitative results are illustrated in Figs. 2–4. As it can be seen, most kinds of error were present at the papers of both domain groups. *This indicates the need for the ITSs to be able to recognise all these different categories of error and provide students with appropriate feedback*.

The comparison of the results of the two groups showed that although most of the error categories were present at the papers of both groups, the percentages of each error category differed between the physics and the economics domain. This means that the categories of error may be

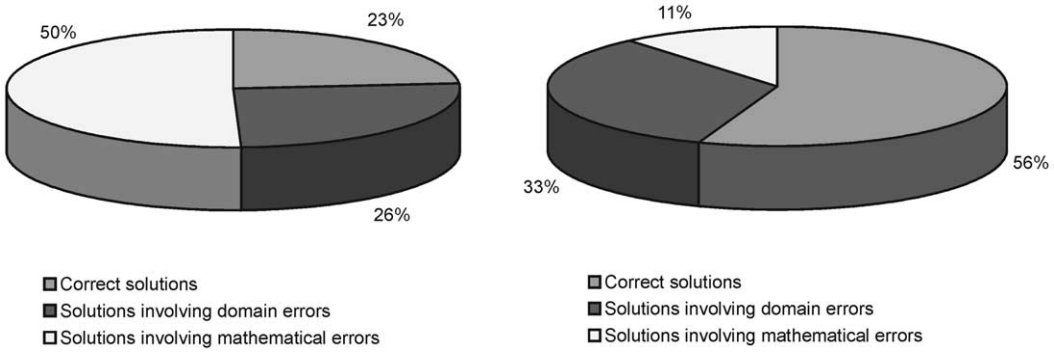

Fig. 2. Distribution of solutions (left: physics group, right: economics group).

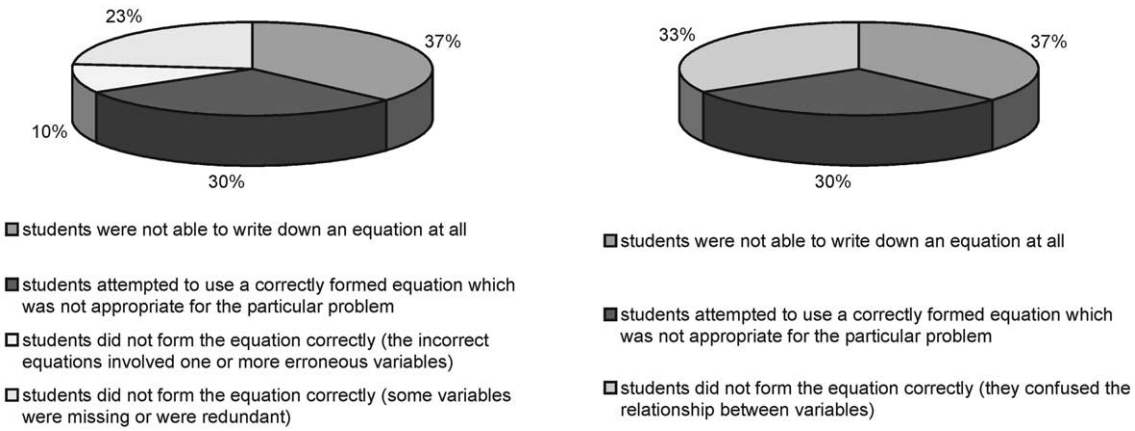

Fig. 3. Distribution of domain errors (left: physics group, right: economics group).

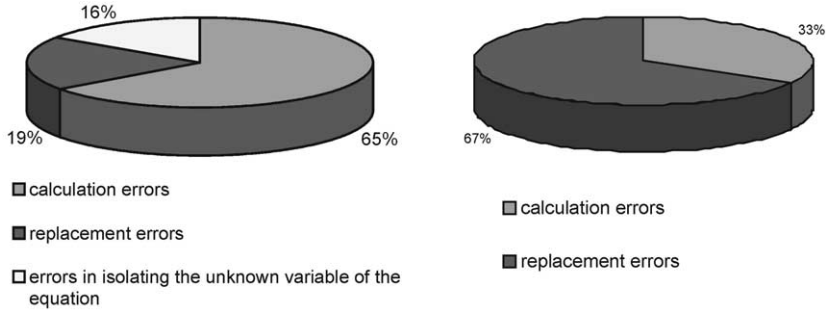

Fig. 4. Distribution of mathematical errors (left: physics group, right: economics group).

the same for different algebra-related domains but the distribution of these errors depends on the particular domain. This is not an unexpected finding: for example, the fact that 50% of the physics problems contained mathematical errors whereas in the economics group these were only the 11%, probably results from the fact that in physics the equations that students should solve are far more complicated than in economics and the values of the variables are usually decimal; therefore, physics students are more prone to mathematical errors. *As a consequence of that, the importance of each kind of error differs between different domains and this affects both the students' grade and the instructional actions that a tutor should perform.*

In our attempt to explore whether several student characteristics could be associated with his/her performance and even further to specific categories of error, we came up with various findings. As expected, students with very high previous year's grades ($\geqslant 18$ out of 20) had a better performance in the test than the rest of the participants. However, looking deeper in the results we came up with some unexpected findings. Concerning the domain errors, 3% were made by students whose previous year's grade in physics was between 18 (out of 20) and 20 (out of 20), 29% by students with grade between 15 and 17, 47% by students with grade between 12 and 14 and only 20% by students with grade less than 12 (Fig. 5). The last two percentages seem quite peculiar but this may be explained by the following hypothesis: Very weak students (grade < 12)

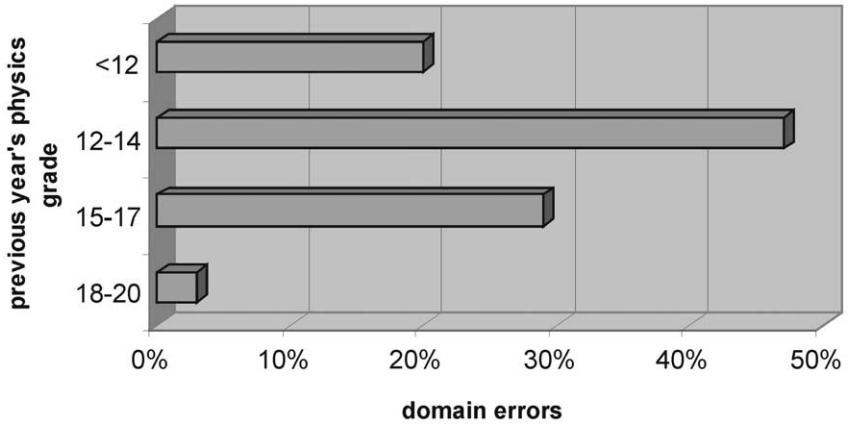

Fig. 5. Previous year's physics grade to domain errors distribution.

are probably trying harder to memorise formulae than students who feel more confident due to their ''not so low'' grade.

In mathematical errors, the relation between the percentage of errors and previous year's grades were different from the case of domain errors. Six percent of the mathematical errors were made by students whose previous year's math grade was between 18 and 20, 31% by students with grade between 15 and 17, 30% by students with grade between 12 and 14 and 32% by students with grade less than 12 (Fig. 6). The surprising result here is that excluding the excellent students (grade > 18), the rest of them contributed almost the same to the total of math errors.

Concerning the correlation between the students' direction and the errors they made we came up with the following findings (Fig. 7):

- Students of the ''science'' direction were responsible for 18% of the total domain errors and 17% of the total mathematical errors.

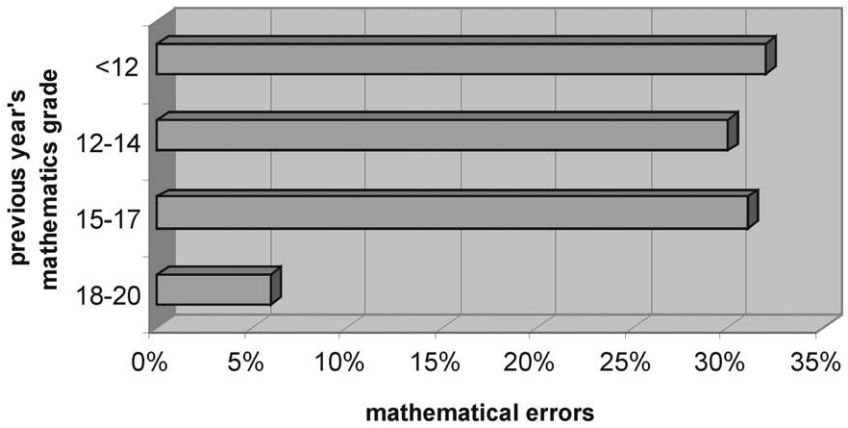

Fig. 6. Previous year's mathematics grade to mathematical errors distribution.

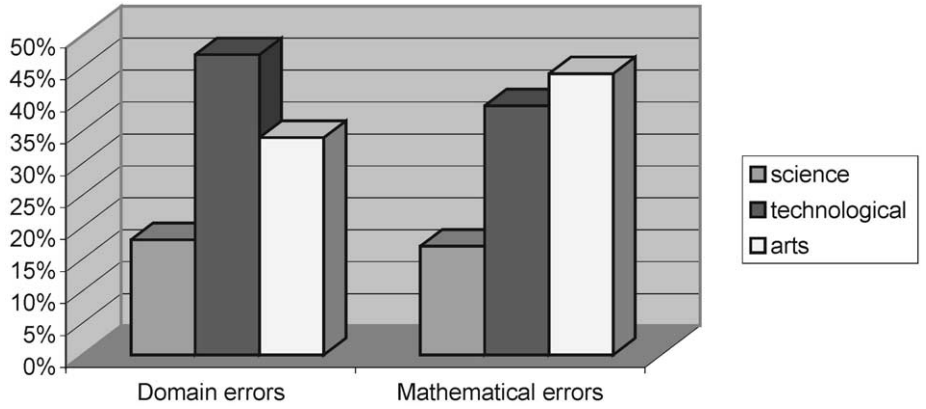

Fig. 7. Correlation between the students' direction and the errors they made.

- Students of the "technological" direction were responsible for 47% of the total domain errors and 39% of the total mathematical errors.
- Students of the "arts" direction were responsible for 34% of the total domain errors and 44% of the total mathematical errors.

As it can be seen from these results, students of the "science" direction performed better and made fewer errors than the rest of the students. This can be attributed on one hand to the fact that this direction of studies attracts mostly the students who have high grades in mathematics and on the other hand to the fact that obviously these students who have chosen such a direction are more willing to deal with physics and math problems. As expected students of the "arts" direction were responsible for the largest percentage of the mathematical errors. However, they were not responsible for the largest percentage of the domain errors. This may be attributed to the fact that these students are not very fond of solving problems in physics or mathematics but they are quite willing to memorise formulae; in this way they are capable of forming the equations correctly but their lack of skills prevents them form solving the problem correctly. The "technological" direction attracts the weakest students of all. Therefore, this group was responsible for the majority of the domain errors. These students made many mathematical errors as well; the underlying reason for this may be that students that think of themselves as being incapable of dealing with mathematics do not choose this direction of studies, since it requires quite a lot of mathematical skills.

Finally, students were asked what aspects concerning their instructor and course they would be interested in knowing before the beginning of a new course. To this, they answered almost unanimously that they were usually asking older students whether their instructor was demanding as for the difficulty of problems s/he gave to them and whether s/he was strict in grading their work.

2.3. Basic operation of WEAR

The aforementioned analysis served as a basis for the design and development of a Web-based authoring tool for the construction of intelligent tutoring systems. The tool is called WEAR,

which stands for WEb-based authoring tool for Algebra-Related domains. An earlier prototype of WEAR is described in Virvou and Moundridou (2000). Most of the study's findings were used in the requirements and design specifications of WEAR. However, some of the study's findings have not been exploited in the current version of the system. These concern the associations of the students' direction of studies and past performance with their current performance. These findings can be utilized in subsequent versions of WEAR or in the design of similar systems. In the remainder of this section we will briefly describe WEAR's operation and highlight the aspects that relate to the common structure of the ITSs that result from the authoring procedure using WEAR.

WEAR provides an authoring environment where instructors may author their ITSs and a learning environment where students may interact with the resulting ITS. In the learning environment, students are presented with a number of problems to work on and are provided with individualised feedback while they are solving them. They also have at their disposal an electronic textbook and are offered navigation support adapted to their individual knowledge. In the authoring environment, the instructor is able to construct problems and tests and author the adaptive electronic textbook. In all cases, WEAR provides automatic assistance, as will be discussed in the subsequent sections.

2.3.1. The authoring environment

The tool takes input from a human instructor about a specific equation-related domain (e.g. economics). This input consists of knowledge about variables, units of measure, formulae and their relation. The instructor does not have to provide the complete list of variables and equations that describe the domain in a single session. S/he may only enter the ones that are needed to solve the problems to be constructed in the current interaction and add more in the interactions to follow. WEAR accumulates domain knowledge each time that the human instructor gives new information. This means that the instructor may provide input to the tool at the same rate as lessons progress in a course. An example of input to the system that an instructor could provide to describe a portion of the domain of economics is shown in Table 1.

WEAR provides the facility of assisting the instructor to create problems where students are asked to give the solution step by step, or multiple-choice tests. When an instructor wishes to create problems, s/he is guided by the system through a step by step procedure. At each step of this procedure the instructor should specify values for some parameters needed to construct a problem. The system follows the instructor's actions and reports any inconsistencies. For example, if the instructor enters values for fewer variables than those needed for the problem to be solvable then the system points out the error. In particular, the procedure of constructing a problem is the following:

- Step 1 (unknown variables): WEAR displays every variable that the instructor has entered when describing the domain (e.g. Table 1) and requests the selection of those that will serve as unknown variables of the problem to be constructed. For example, an instructor may have selected GDP—Gross Domestic Product and NX—Net exports to be the unknown variables of a problem.
- Step 2 (known variables): The instructor is requested to select the variables that will serve as the given data of the problem. For example, s/he may have selected GNP—Gross

Table 1
Input example from the domain of economics

Variable's description	Variable's name
Gross Domestic Product	GDP
Gross National Product	GNP
Net Factor Payments from abroad	NFP
Private Consumption	C
Investment	I
Government consumption and investment	G
Net exports	NX
Private disposable income	DY
Transfers received from the Government	TR
Interest payments on the Government Debt	INT
Taxes paid to the Government	T
Private saving	Spvt
Government saving	Sgovt
National saving	S
Current account balance	CA
Equations	
GDP = GNP-NFP	Spvt = DY-C
GDP = C + I + G + NX	Sgovt = T-TR-INT-G
DY = GDP + NFP + TR + INT-T	S = Spvt + Sgovt
CA = NX + NFP	S = I + CA

National Product, NFP—Net Factor Payments from abroad, I—Investment and S—National saving to be the known variables of the problem.

- Step 3 (values of the known variables): The instructor is requested to enter values for the known variables of the problem.
- Step 4 (verification of WEAR's choices): At this step (Fig. 8) the system presents to the instructor a simple problem text that it has produced which describes the given and asked data. The instructor may change this text to make it more realistic and comprehensible. The information concerning the known and unknown variables is used by WEAR to examine the domain equations and isolate the ones that are needed for the problem to be solved. In some domains more than one equation exists that defines the same variable. In such cases WEAR finds all the relevant equations to the variables selected. Then it presents them to the instructor who is expected to select the appropriate ones for the problem s/he is currently constructing.
- Step 5 (problem's name and difficulty level): The instructor is requested to enter a name for the problem s/he has constructed and to define the problem's level of difficulty.

Finally, WEAR allows the authoring of electronic textbooks by instructors and delivers them over the WWW to learners (Moundridou & Virvou, 2001a). These textbooks offer navigation support to students, adapted to their individual needs and knowledge. The authoring procedure to create an adaptive electronic textbook with WEAR is quite simple. In particular, the instructor should prepare HTML files for the topics that would be contained in the electronic textbook. The

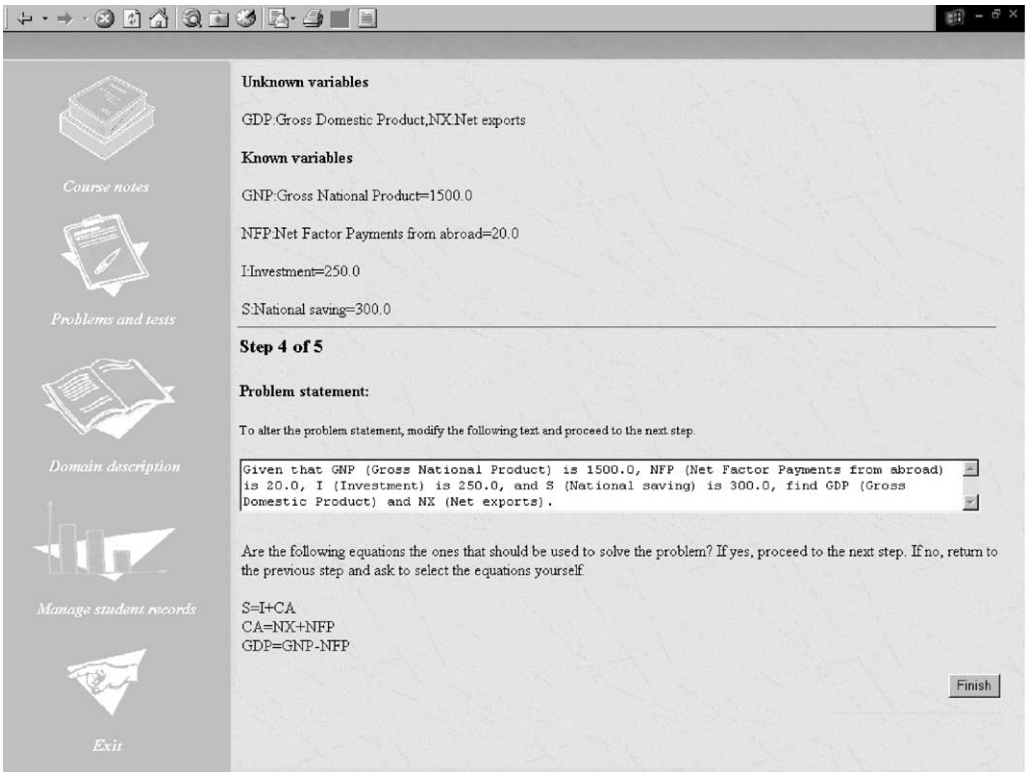

Fig. 8. Problem construction (step 4 of 5).

next step is to use WEAR's facilities for uploading these files to the WEAR server. For each uploaded file the instructor must specify a title, a difficulty level and the position that it should have in the topics hierarchy. S/he should also relate topics to the domain variables and edit the *is_prerequisite_of* and *is_related_to* relationships between topics.

2.3.2. The learning environment

The knowledge representation of the domain being taught, as well as information obtained from student models, are exploited by WEAR to provide adaptive navigation support to students (Brusilovsky, 1996). To achieve this, WEAR makes use of the adaptive link annotation technique: students interacting with the system see visual cues (different icons next to each link) that inform them about the current state both of the available problems and of the topics constituting the teaching material. This is done in order to facilitate the student's choice about which problem to solve next and which topic to study, as well as to provide them with information concerning the already mastered topics and concepts.

When a student attempts to solve a problem, the system provides an environment where the student gives the solution step by step. At first the student is presented with a problem statement like the one shown in Fig. 9. The student is requested to write down the equations that are needed to solve the problem and then s/he is requested to mathematically solve the problem. To detect the erroneous answers the system compares the student's solution to its own at every step. The

Fig. 9. Solving a problem while in student's mode.

system's solution is generated by WEAR's Problem Solver, which is implemented in PROLOG. The Problem Solver incorporates knowledge about how to solve systems of linear equations correctly and may generate the solution to a problem using information about the specific domain to which the problem belongs (e.g. physics). During the process of solving a problem the student's actions are monitored by the system. In case of an erroneous action, the Problem Solver passes the student's answer to the Student Modeller, which is then responsible for diagnosing the cause of the error. The errors that are recognised by WEAR's Student Modeller are these that the empirical study indicated that exist:

Domain errors. These include errors that are due to the student's unfamiliarity with the domain being taught. For example, if a student enters the equation $S = I - \mathrm{CA}$ instead of the correct $S = I + \mathrm{CA}$, then the error is attributed to the category of Domain errors and in particular to the sub-category of ''erroneous relationship between variables''.

Mathematical errors. These include errors that are due to the student's lack of skills in solving mathematical equations. Such errors could be calculation errors, errors in isolating the unknown variable, etc. For example, if a student trying to isolate CA in the equation $S = I + \mathrm{CA}$ enters $\mathrm{CA} = S + I$ instead of $\mathrm{CA} = S - I$, then the error is attributed to the category of Mathematical errors and in particular to the sub-category of ''wrong isolation of the unknown variable''.

The Student Modeller performs error diagnosis based on the above categories of error and then the system provides the student with the appropriate feedback message indicating the exact error that the student has made. If for example the student has entered the erroneous answer CA = 150000 instead of the correct CA = 50000, the feedback message that s/he will receive will be: "Your current response **CA = 150 000** is wrong. Check the calculations you have made." The student interface includes an animated speaking character, which is responsible for communicating the instructions and feedback messages to the students.

The information concerning the particular kinds of error that each student makes is kept in the student model and is used by WEAR in order (1) to resolve ambiguities that arise from errors for which more than one hypothesis can be generated as to what the cause of the error has been and (2) to form individualised detailed progress reports of the student, which could be requested by the student and/or the instructor.

3. Experiment concerning the authoring procedure

3.1. Aims and settings of the experiment

The second and most extensive part of our experiment involved instructors only. The main aim of this part of the experiment was to identify issues that related to the instructors' needs and expectations when they were building an ITS using an authoring tool. In that way, we would be able to decide upon the final form that each component of the generated ITSs should have and most importantly upon the way the authoring of the ITSs should be performed and supported.

This part of the experiment involved six instructors. The six instructors were different from those involved in the first part. They all had a first and/or higher degree in Physics or Mathematics and were all experienced as classroom instructors. Half of them had more than 10 years teaching experience and the rest of them from 3 to 5 years. Finally, we should note that all of the instructors were very competent computer users.

The six instructors were given a description of the first part of the experiment along with the results of it and were asked to categorise the most frequent student errors and associate each of them to a stereotype of student. Furthermore, we asked them to grade several student papers and also to define the difficulty level of each exercise that was given to students. Instructors were also asked to answer some questions concerning their teaching style and preferences. All these were done so that we could collect information about the instructors' attitude towards teaching and students' learning in real settings. This information would lead to design decisions concerning the following issues: (1) how the generated by the authoring tool ITS should be acting as a virtual instructor and (2) what parts of the ITS should be parameterised for the instructors to be allowed to define their own structure or behaviour.

3.2. Analysis of the results

Based on the analysis of the information collected from the instructors in this part of the experiment, we reached a general conclusion. This was that the instructors' attitude toward the teaching/learning process varied a lot among them. As will be discussed in the subsequent

paragraphs, different instructors associated differently student errors to student stereotypes; they also gave different weight to the same student errors, they graded student papers different and finally the level of difficulty they assigned to each problem was in some cases very dissimilar.

In particular, the levels of difficulty (ranging from very easy to very difficult) that instructors assigned to five physics problems were in some cases very diverse (e.g. for a particular problem half of the instructors stated that it was difficult and the other half that it was very easy). Based on this result, we can assume that there are cases when an instructor may overestimate or underestimate the level of difficulty of the problems s/he poses to his/her class. In a real setting the instructor is able to explore immediately the results of his/her teaching and reconsider his/her opinion. *To be able to do the same (adjust his/her teaching decisions, such as the difficulty level of problems) in an e-learning scenario, the instructor should be offered feedback by the authoring tool concerning the performance of his/her class.*

Instructors were also asked to state for each kind of error the frequency it occurred in real settings and to give a weight for its importance. The instructors' answers indicated that there were errors for which instructors almost agreed on their frequency and importance and others for which they completely disagreed. For example, for the error that students make when they attempt to use a correctly formed equation which is not appropriate for the particular problem, almost half of the instructors stated that this happens rarely, whereas the other half stated that this is a frequent mistake. Accordingly, some of the instructors considered this kind of error to be very important and some others to be not so important. Twidale (1992) reports on a similar finding and notes that the differing importance attached by different teachers to certain errors affects the degree to which they are likely to intervene in a student's action. He also argues that there appear to be at least three different ways that teachers can classify errors: the complexity of the concept, the difficulty of learning the correct rule and the personal preference of the teacher. Based on this, and on our observations we can argue that *instructors when authoring an ITS should be allowed to have the responsibility for defining the importance of each kind of error since this affects the students' marks and reflects the instructors' personal teaching style.*

From the instructors' answers to questions concerning their teaching style and preferences we came up with the finding that their attitude towards their students is sometimes based on certain students' characteristics and not only on the students' current and actual performance. For example, some instructors stated that the grade they gave to a student depended on the effort that s/he had made; some others stated that they were more strict in grading a student that repeatedly made the same mistake or less strict with novice or careless students. Thus, *an ITS authoring tool should monitor students closely, collect observable information of this kind and ask the instructor to specify which parameters should be used to grade students.*

Another interesting finding was that instructors usually looked deeply into their students' solutions to problems and they did not just check if their final answer was correct or not. They did so in order to discover possible misconceptions that students may have had, especially in the cases of difficult problems and/or weak students. Accordingly, instructors stated that if their students were working with a computer program to solve problems *they would like to be provided with detailed information concerning the students' performance.*

The experiment also revealed a great diversity of instructors' opinions about the underlying cause of students' errors. This was also the case with the course of action that each instructor

would follow to help the student correct his/her erroneous answer. For example, instructors were asked what their advice would be to a student if s/he wrote down a correct equation which was not appropriate for the particular problem. Some of the different instructors' responses were the following:

"I would suggest him to read again and more carefully the problem statement",
"I would tell him to write down every equation s/he knows and reject the ones that s/he considers not appropriate for the specific problem",
"I would point out the existence of an error and would ask the student to find out how s/he could fix this problem",

The diversity of instructors' beliefs and actions concerning the errors that students make, leads to the following important design decision: *instructors should be allowed the responsibility for specifying how they wish the ITS to respond to each kind of student error*.

The first part of the experiment revealed the fact that students were interested in knowing some aspects concerning their instructor. In view of this, instructors were asked whether they were aware of this. Most of them responded positively and described several situations of this kind. For example, an instructor said: "Students are indeed interested in knowing several things about their instructor. Preconceptions of this kind affects students' idea about the course they are going to have. However, not all students are being affected in the same way: good students are attracted by a demanding instructor who motivates them to try harder; on the contrary, weak students are discouraged when their instructor is demanding." *To allow students to have a more personal interaction with the ITS, they should be allowed to know about some characteristics of their instructor that have either been inferred by the system or have been explicitly stated by the instructor*.

3.3. Incorporating an instructor modelling component in WEAR

The most important finding of the study that we conducted was the following: ITS authoring tools should allow instructors to put their own teaching style and knowledge into practice. To maximise the flexibility and adaptivity of WEAR to instructors, we incorporated an instructor modelling component into its architecture. The resulting architecture and design refinements of WEAR are described in the following subsections.

3.3.1. System's architecture

The system's underlying architecture is shown in Fig. 10. The *Authoring components* contain the system's modules dealing with courseware construction and management. These are tools for describing a domain in terms of variables and equations, associating domain variables with topics of the electronic textbook, specifying relationships between topics, uploading teaching material, managing student records, constructing new problems and tests and retrieving problems that were previously constructed. The information that the *Authoring components* receive from the instructor is used for the construction of *Domain knowledge and problems*.

The *Instructor modeller* is responsible for building and updating each instructor model. The *Instructor model* holds: (1) information obtained explicitly by asking the instructors (such information may be the instructor's preferences concerning the course and his/her teaching expertise),

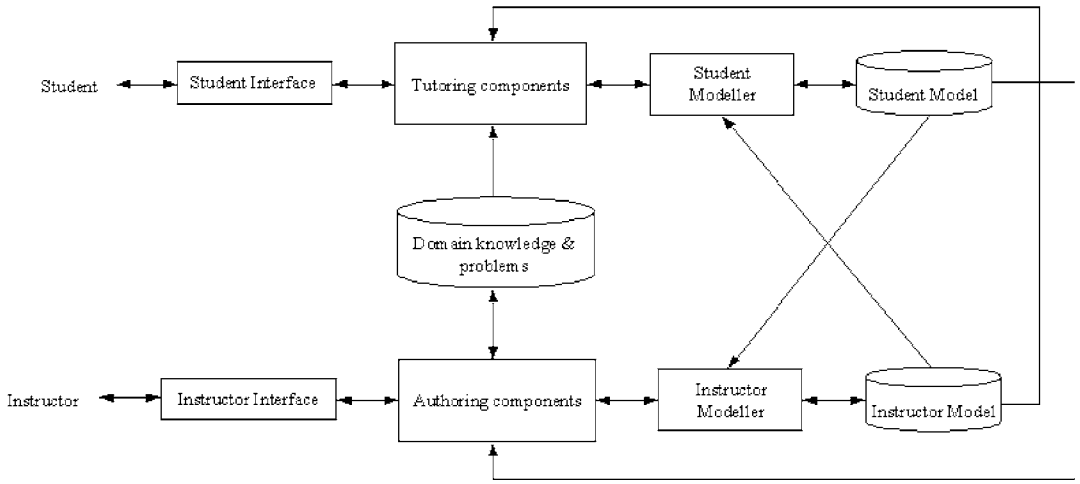

Fig. 10. WEAR's architecture.

and (2) implicit information inferred by WEAR (such as the instructor's interest in some categories of problem). The *Instructor model* provides information that is used by the system to individualise the interaction with each instructor; for example, when an instructor browses the categories of problem, the ones that s/he is interested in are already pre-selected by the system.

The *Tutoring components* consist of components that interact with students while they are solving problems, present the teaching material in an adaptive way and form individualised advice for students. To perform these tasks, the *Tutoring components* need to know who each student is and what s/he knows so far. In addition, the *Tutoring components* need to know what the structure of the domain being taught is (e.g. what the prerequisite concepts of each concept are) and what the correct equations that describe this domain are. The sources for all this information are the *Student models* and the *Domain knowledge and problems*. *Student models* are updated by the *Student modeller* each time a student interacts with the system. When the student makes a mistake, the *Student modeller* is responsible for diagnosing the cause of it.

The resulting ITSs from WEAR have a *Student interface* that operates in two modes: the instructor and the co-student. In both cases there is a speaking animated cartoon-style agent which communicates the system's messages to the student. The underlying reasoning used for both modes is based on the *Student model* and *Instructor model*. The difference of these modes is restricted to the interface level. In particular, in the instructor mode, the interface agent simulates an instructor and provides rather formal messages to the student whereas in the co-student mode, the agent simulates a co-student that provides messages in a more casual way. On the other hand, the *Instructor interface* does not include any animated agent and it operates as a conventional GUI.

As shown in Fig. 10, WEAR uses its instructor and student model for instructors and students, respectively, but also vice versa. This means that the model of each class of user is also used as a source of information to be passed to the other class of user. This is done both explicitly by informing the other class of user and implicitly by affecting the model of the other class of user. For example, instructors may be given some information about students' performance from the student model in order for instructors to evaluate their own teaching strategy. Similarly, students may be given explicitly some information about their instructors' preferences concerning their

teaching strategies. In this way students may have a better idea about how to collaborate with their instructor.

The above examples concern the case when each class of user may be given explicitly some information from the model of the other class. However, most importantly the models of the two classes of user interact with each other and affect the modelling process itself. For example, the students' performance recorded in the student models is used to calculate the degree of an instructor's tendency to overestimate or underestimate the level of difficulty that s/he assigns to problems. If a high degree of such a tendency seems to exist, it is recorded in the instructor's model and used to provide individualised help to him/her (e.g. to remind the instructor of this tendency when s/he constructs new problems). Similarly an instructor model may affect student models. For example, the students' level of knowledge, which is recorded in student models, is assessed taking into account the students' errors. These may either be mathematical or domain errors. By default WEAR considers the two kinds of error equally important; however, if an instructor model indicates a specific instructor's preference to weigh more one kind of error than the other, then the students' level of knowledge is calculated taking into account the instructor's preference.

The implementation of the system is based on the client–server architecture. WEAR resides on a Web server. Both students and instructors are clients who can use the teaching and authoring services offered by the system using a conventional Web browser.

3.3.2. Additional instructors' facilities in authoring

Instructors are requested to classify themselves in one of three categories concerning their teaching expertise: ''novice'', ''having little experience'', and ''experienced''. This information is stored in their instructor model and used by WEAR to provide them with adapted help concerning the selection of teaching strategies. Moreover, instructors are asked to explicitly provide their long-term preferences for the course they author in terms of the course's difficulty and popularity. WEAR compares these preferences, which are also part of the instructor's model, with the instructor's actions. If an inconsistency is present, WEAR informs the instructor that the course does not seem to follow his/her long-term goals. For example, an inconsistency between the instructor's actions and his/her long-term goals may arise when s/he has stated that s/he wishes the course to be easy and the problems s/he has constructed are difficult.

While students are working on the given problems the system collects evidence about the level of difficulty so that it can provide feedback to the instructor. For example, if the majority of the students of a certain level have failed in solving a particular problem, which has been assigned to this level, then the instructor is informed. In a case like this, perhaps the instructor may wish to reconsider the level of difficulty since there is evidence that the problem may be of a higher level of difficulty. On the other hand, if many students have managed to solve a problem of a higher level of difficulty than the one proposed by the instructor, the level of difficulty may have been overestimated by the instructor. In this case too, the system informs the instructor. In both cases, the tool does not take the initiative to alter the level of difficulty by itself: it suggests the instructor to increase or decrease this measure according to the observed students' performance in a specific problem. In this way an instructor is being assisted by the system in the classification of problems.

Beyond constructing a problem by himself/herself, the instructor has the ability to explore the problems constructed by others and choose the ones that s/he desires to be accessible by his/her

class. Every time an instructor constructs a new problem the system performs this problem's categorisation based on some parameters. The problems are first categorised according to the domain to which they belong. At a second level the problems of each domain are categorised according to the variables they involve and their level of difficulty. Every variable of the domain can possibly form a problem category. For example, a problem like: ''A force of 100 Newtons is acting on a 25 kg object which is initially stable. After 10 secs how much is the impulse?'' belongs to the broad category ''Physics'' and in the sub-categories ''Impulse'', ''Velocity'' and ''Acceleration'' due to the variables involved in it. The same problem could also belong to the sub-category ''level of difficulty 1'' based on the problem's level of difficulty as this has been defined by the instructor.

Instructors are allowed either to browse the collection of problems by selecting the categories and sub-categories that match their needs and interests, or to search the entire collection using some keywords. The instructor modelling mechanism is responsible for tailoring the interaction of the instructors with the system to the instructors' needs.

Finally, when building an electronic textbook, instructors are provided with tools that verify the consistency of the course and report possible problems or errors, such as the case when the prerequisite relationships imply that a topic indirectly requires the knowledge of itself. To offer more intelligent and individualised help WEAR relies on the information provided by the instructor modelling component that it embodies.

3.3.3. Instructor and student models and their interaction

The instructor modelling component monitors each instructor's interactions with WEAR and constructs and/or updates his/her user model. In particular, the instructor aspects that are being modelled in WEAR, are the instructor's preferences, usual activities, special interests and his/her level of expertise in teaching. These are described below.

In WEAR the instructor may give some long-term preferences as to whether s/he wishes the course to be difficult, average or easy or whether s/he wishes it to be very popular or fairly popular or whether s/he is not interested in this feature. Each of these preferences is associated with student interest in the course and with a percentage of failure in performances of class students. The student interest in the course is measured by observable actions of students, such as how many times students visit the electronic textbook, and/or how many problems they have solved. The instructor may also state how important s/he considers each category of student error to be. In that way the students' level of knowledge could be calculated according to the instructor's preferences, assigning higher weight to those errors that the instructor has defined as more important.

Instructor's activities that are frequent are also recorded in his/her long-term model. For example, if an instructor often constructs problems that belong to the same category then it is inferred that this particular instructor is a ''major contributor'' in that sort of problem and this usual activity of his/her is recorded in his/her user model. In addition, the instructor's interests are inferred and recorded in the long-term instructor model. For example, if an instructor frequently searches for specific categories of problem then the inference made is that this instructor is ''interested'' in these categories of problem and this special interest of his/her is recorded in his/her user model.

Finally, in WEAR the instructor model records the teaching expertise of the instructor. This is explicitly stated by the instructor himself/herself. Each instructor may situate himself/herself in

one of three categories: novice, having little experience and experienced. In the case of novice tutors and those having little experience, the authoring tool offers more detailed help concerning the teaching strategies that the tutor may select and shows him/her by default the results of the consistency checks.

The instructor model is utilised by the system in the following ways:

To provide individualised help to the instructor. For example, if an instructor has stated a long-term goal that s/he wishes to render the course popular within the class students then the authoring tool will examine whether the instructor's long-term goal is far or not from being met. Student models provide information about how many students have attempted certain exercises and how many times they have seen certain lectures.

To adapt the interaction with instructors. When an instructor wishes to find a problem and decides to browse the available categories, s/he will see that in the categories' list the ones that s/he frequently explores are pre-selected for him/her by the system. In addition, if new problems belonging to the categories that a particular user is interested in are added, the system informs the user when s/he logs in.

To promote collaborative work among instructors. Users are offered the choice of seeing what other users have done along two dimensions: the course structure and the constructed problems. Concerning the former, the information that is presented to the instructor is the structure of a similar course created by another instructor. In that way, instructors who may be novice as course designers could be assisted by more experienced peers who have previously used WEAR. When selecting to see problems constructed by others, the instructor is presented with a list of problems constructed by instructors who are considered by the system as ''major contributors'' in the categories that this specific instructor is considered ''interested''.

To provide a more personalised learning environment for the student. As has been already mentioned, the interface with the student works through a speaking agent and it operates in two modes: the instructor mode and the co-student mode. In both cases of the instructor and the co-student mode, the instructor model is used to pass some information about the instructor to the student. For example, as the empirical study has shown, students would like to know to some extent who their instructor is; what the instructor expects from the students or what his/her priorities are in terms of the course taught, how difficult the exercises s/he gives are, etc. Usually, they seek such information from older students who had the same instructor in the past, or they ask such questions to their instructors themselves. However, such information may be passed to them from the instructor modelling component through the student interface. This is done in order to render the interaction with the ITS more personalised for the student. Therefore, the virtual co-student uses information from the instructor modelling component to answer questions of students that concern certain aspects of their instructors, e.g. what the success rate was in the previous course given or whether the course is considered by the instructor as hard or not. However, these pieces of information are passed to the student only if the instructor has agreed to this.

The student model that WEAR maintains is a combination of a stereotype and an overlay student model, similarly with other systems such as (Hohl, Böcker, & Gunzenhäuser, 1996). The stereotype student model is formed either directly by the instructor or after a preliminary test that

has been posed to the student. It classifies initially the student according to his/her knowledge of the domain and his/her mathematical skills. As a result of this, each student is assigned to a stereotype (novice, beginner, intermediate or expert). The stereotype model defines initial values for the overlay student model. The latter is represented by a set of pairs "concept-value". The concepts are domain concepts and concepts concerning the equation solving process (e.g. isolating the unknown variable in an equation). Domain concepts include domain variables and topics constituting the teaching material. For example, each variable presented in Table 1 constitutes a domain concept for the economics domain. The value for each concept is an estimation of the student's knowledge level of this concept and it is initialised by the stereotype student model. If, for example, the stereotype model indicates that a student is "intermediateø as to his/her mathematical skills and "beginner" as to his/her knowledge in the domain, then the concepts constituting the overlay student model are given values according to the following assumptions: every concept that concerns the equation solving process and that has not been rated by the instructor as difficult or very difficult is considered known by the student; every domain concept rated as very easy is considered already known. After the initialisation of each "concept-value" pair, the student model is updated taking into account the student's performance in solving the problems associated with this concept and the reading or not of the corresponding teaching material. For example, if a student has successfully solved all problems evaluating the domain concept "Gross Domestic Product—GDP" and s/he has also read the corresponding topics of the electronic textbook, then in his/her student model the concept "GDP" will hold the value 1 and thus it will be considered known.

4. Evaluation of the design

The last phase of the study was an evaluation of the design and preceded the final implementation of the system which has now been completed. This phase involved both instructors and students and aimed at verifying the design guidelines of both the authoring and learning environments of WEAR.

The instructors who participated in this part of the study were given a description of WEAR and were asked to rate its functions and utilities by assigning to each of them a value between 1 and 5. In particular, 5 stood for "extremely useful", 4 for "useful", 3 for "probably useful", 2 for "probably useless" and 1 for "completely useless". The described functions were reflecting the resulting from the first two parts of the study design guidelines. The mean ratings for each function/utility (in order from the function with the highest to the one with the lowest rating) are shown in Table 2.

These results show that the majority of the instructors liked the functionality of the system that concerned the construction of problems and the problem solving support provided to students (rating $\geqslant 3.5$). On the other hand, instructors were more restrained in their ratings when it came to utilities that would help them collaborate with peers. However, what seems to be important and rather encouraging is that all the described functions were considered by the instructors as at least "probably useful" (the lowest mean rating was 2.8, while 3 stood for "probably useful").

Students' involvement in this verification phase of the study, concerned the evaluation of the user interface component of the generated by WEAR ITSs. We restricted this part of the investigation

Table 2
Means of instructors' ratings for the functionality of a hypothetical Web-based ITS authoring tool

Functions/utilities	Mean rating
You can specify the level of difficulty for each problem you are constructing. Based on this and on the student's level of knowledge, the tool provides advice to students concerning which problems they should try to solve	4.8
The tool allows you to provide problems like the ones described in this experiment as well as other teaching material	4.5
The tool assists you in the problem construction phase (checks whether the problem is solvable, produces the solution, etc.)	4.3
The tool allows you to define the weight of each kind of error in the students' grade	4.2
Students are being modelled by the system and you can examine at any time the values that the attributes of these student models hold	4.2
Students can solve problems with this tool which performs error diagnosis and provides each student with individualised feedback	4.0
You can register your students in a virtual class	3.7
You can associate topics of the curriculum with domain concepts and define prerequisite relationships between them. Based on this information the tool suggests to each student which topic s/he should study each time depending on his/her current level of knowledge	3.5
The tool informs you in case it has inferred (based on your students' performance) that you have the tendency to over or underestimate the level of difficulty that you assign to problems	3.5
You are able to search for problems constructed by other instructors who are using the tool and adopt problems you find useful for your own virtual class	3.2
The interface of the tool in the student's mode of function incorporates an animated speaking character that represents a virtual instructor and is responsible for communicating the system's messages to the student	3.0
The tool infers your interests and maintains a model for you. Based on your model the tool informs you when new problems that might interest you are added in the database by other instructors	2.8

to the approval of the user interface since it is not possible for students to evaluate the efficiency of the system as a learning aid or to check whether the error diagnosis it performs is valid or not. However, the students' opinion for the interface of a learning environment is of great importance since it is an important factor for the acceptance or not of the tool in real settings. In particular, two kinds of experiment were performed, each with a different interface agent embodied in the interface of the ITS (Moundridou & Virvou, 2001, 2002).

For the first experiment a speech-driven anthropomorphic agent was incorporated in the student interface and was responsible for guiding students in the environment and communicating the system's feedback messages. Two groups of students each consisting of 24 subjects worked either with this version of the ITS or with an agent-less version of it. Through data collected from log files, questionnaires, pre- and post-tests, we reached the conclusion that there were mainly two advantages that were induced by the presence of the agent. The first advantage concerned the enjoyment that students felt when they interacted with a system that embodied a speaking animated interface agent. The other advantage was that students working with the agent version of the system found the problems that they should solve less difficult than students working without

the agent did, despite the fact that the performance of both groups of students was similar. The importance of this finding is great concerning the motivation of students to work with the system; it shows that students working with the agent version are more motivated than students working with the agent-less version are.

After conducting a short investigation study concerning the type of animated agent that we should use, we replaced the anthropomorphic agent with a cartoon character. The investigation that we conducted was among students of various ages and backgrounds. Among other questions we also asked them what type of animated agent they would prefer in an interface. The majority of students seemed to have favoured a cartoon character rather than an anthropomorphic one. The main reasons for this was that they considered cartoon characters more pleasant and cute and less stressful. As a consequence, the new speech-driven animated agent was Microsoft's Peedy the parrot. A second experiment similar to the one mentioned above was conducted to verify if the previous positive findings still existed. The results showed that this was true. Indeed, students working with Peedy seemed even more enthusiastic as their enjoyment ratings indicated.

Conclusions and future work

In the last decade a large number of researchers in the area of computer-based learning have put their efforts in the design and development of authoring tools for intelligent tutoring systems. The reason for the increased interest in authoring tools is that they provide environments where instructors may author their own ITSs reflecting exactly their own teaching style and preferences. However, the construction of an authoring tool inherits the difficulty of the construction of an ITS and it is even more complex since authoring tools should operate effectively both for the instructors who intend to build an ITS and for the students who should learn from it. To design a successful and effective ITS authoring tool one must perform a careful and extensive requirements analysis in which both classes of prospective users (instructors and students) should be involved. This is even more the case when the authoring tool is meant to operate over the Web.

In this paper we reported on an empirical study that we conducted in order to design and develop WEAR, a Web-based ITS authoring tool for algebra-related domains. In the three phases of the study we investigated several aspects concerning the attitude and behaviour of both students and instructors. This effort proved to be really useful in defining the design guidelines of WEAR's authoring and learning environments. Some of the findings were domain-dependent in the sense that they could apply to algebra-related domains. This means that these findings could be useful for ITSs and ITS authoring tools similar to WEAR. However, a lot of the findings were absolutely domain-independent and therefore could be useful for other authoring tools as well. Such findings refer mainly to the instructors and students' attitudes and could be used in authoring tools to enhance their functionality.

In particular, beyond the practical issues that we explored (e.g. kinds of students' errors that the ITS should recognise), the emphasis was given to the exploration of the instructors' beliefs and actions. This gave us insight for specific design guidelines concerning the authoring procedure. The fact that most aspects of the teaching/learning process are viewed by different instructors in different ways led to the important requirement of incorporating an instructor modelling component in the architecture of WEAR. This requirement is completely domain-independent and therefore

could be used for any authoring tool. Indeed, an instructor modelling component may render the system more flexible and adaptable to particular instructors' interests and needs. The instructor modelling component interacts with other components of the system in order to form an instructor model which is used for tailoring advice to the individual instructor. Furthermore, the study also revealed the need for enriching the role of instructors by providing them with relevant information throughout the ITS development life cycle.

The incorporation of an instructor modelling component in the architecture of an ITS authoring tool is a novelty in the area of authoring tools. WEAR's paradigm proves the usefulness of this approach and reveals several aspects that are open for further research. For example, the exact information that an instructor model should hold, the sources for acquiring such information and the way that the instructor models could be utilised are issues that should be explored further. The results of the research in the area of user modelling can definitely provide useful guidelines for the instructor modelling approach. Various user modelling methods and techniques (e.g. Carberry, 2001; Webb, Pazzanim & Billsus, 2001; Zukerman & Albrecht, 2001) can be applied to model the instructors in ITS authoring tools and to use these models to improve the quality of such systems.

Finally, the most important asset of WEAR is the fact that its design was based on the results of an extensive and multilateral investigation. The dearth of reports on knowledge acquisition for ITSs is a fact that has already been acknowledged in the literature (e.g. Twidale, 1992). This is even more the case for authoring tools where such reports are completely absent. Reporting on studies like the one described in this paper is by itself calls for research in the area of ITSs and ITS authoring tools since they can assist in the accumulation of knowledge to guide the development of future systems.

References

Alpert, S. R., Singley, M. K., & Fairweather, P. G. (1999). Deploying intelligent tutors on the web: an architecture and an example. *international journal of artificial intelligence in education*, *10*(2), 183–197.

Bell, J., & Hardiman, R. J. (1989). The third role—the naturalistic knowledge engineer. In D. Diaper (Ed.), *Knowledge elicitation: principles, techniques, and applications* (pp. 49–85). Chichester, England: Ellis Horwood Ltd.

Berz, M., Erdelyi, B., & Hoefkens, J. (1999). Experiences with interactive remote graduate instruction in beam physics. *Journal of Interactive Learning Research*, *10*(1), 49–58.

Boose, J. H. (1993). A survey of knowledge acquisition techniques and tools. In B. G. Buchanan, & D. C. Wilkins (Eds.), *Readings in knowledge acquisition and learning* (pp. 29–56). San Mateo, CA: Morgan Kaufmann.

Brusilovsky, P. (1996). Methods and techniques of adaptive hypermedia. *User Modeling and User-Adapted Interaction*, *6*(2–3), 87–129.

Carberry, S. (2001). Techniques for plan recognition. *User Modeling and User-Adapted Interaction*, *11*(1–2), 31–48.

Chou, C. (1999). Developing CLUE: a formative evaluation system for computer network learning courseware. *Journal of Interactive Learning Research*, *10*(2), 179–193.

Collins, A. (1992). Toward a design science of education. In E. Scanlon, & T. O'Shea (Eds.), *New directions in educational technology* (pp. 15–22). Berlin: Springer.

Corbett, A. T., & Anderson, J. R. (1992). LISP intelligent tutoring system: research in skill acquisition. In Larkin, J. H. & Chabay, R.W. (Eds.), *Computer-assisted instruction and intelligent tutoring systems: shared goals and complementary approaches* (pp. 73–109). Hillsdale, NJ: Lawrence Erlbaum Associates.

Eliot, C., Neiman, D., & Lamar, M. (1997). Medtec: a web-based intelligent tutor for basic anatomy. In S. Lobodzinski, & I. Tomek (Eds.), *Proceedings of WebNet '97, World Conference of the WWW, Internet and Intranet* (pp. 161–165). Charlottesville, VA: AACE.

Garg-Janardan, C., & Salvendy, G. (1988). A structured knowledge elicitation methodology for building expert systems. *International Journal of Man-Machine Studies*, *29*(4), 377–406.

Hendler, J., & Feigenbaum, E. A. (2001). Knowledge is power: the semantic web vision. In N. Zhong, Y. Yao, Liu, & S. Ohsuga (Eds.), *Web Intelligence: Research and Development, Proceedings of the First Asia-Pacific Conference, WI 2001, lecture notes in artificial intelligence, vol. 2198* (pp. 18–29). Springer: Berlin.

Hohl, H., Böcker, H., & Gunzenhäuser, R. (1996). Hypadapter: an adaptive hypertext system for exploratory learning and programming. *User Modeling and User Adapted Interaction*, *6*(2–3), 131–155.

Kitto, C. M., & Boose, J. H. (1989). Selecting knowledge acquisition tools and strategies based on application characteristics. *International Journal of Man-Machine Studies*, *31*(2), 149–160.

Koedinger, K. R., Anderson, J. R., Hadley, W., & Mark, M. (1997). Intelligent tutoring goes to school in the Big City. *International Journal of Artificial Intelligence in Education*, *8*, 30–43.

Lajoie, S.P. (1993). Computer environments as cognitive tools for enhancing learning. In S. P. Lajoie, & S. J. Derry, (Eds.), *Computers as cognitive tools* (pp. 261–288). Hillsdale, NJ: Lawrence Erlbaum Associates.

Moundridou, M., & Virvou, M. (2001a). Authoring and delivering adaptive web-based textbooks using WEAR. In T. Okamoto, R. Hartley, Kinshuk, & J. P. Klus (Eds.), *IEEE International Conference on Advanced Learning Technologies; Issues, Achievements, and Challenges—ICALT 2001* (pp. 185–188). Los Alamitos, California: IEEE Computer Society.

Moundridou, M., & Virvou, M. (2001b). Evaluating the impact of interface agents in an intelligent tutoring systems authoring tool. In N. Avouris, & N. Fakotakis (Eds.), *Advances in human-computer interaction I: Proceedings of the Panhellenic Conference with International participation in Human-Computer interaction—PC-HCI 2001* (pp. 371–376). Patras: Typorama Publications.

Moundridou, M., & Virvou, M. (2002). Evaluating the persona effect of an interface agent in a tutoring system. *Journal of Computer Assisted Learning*, *18*(3), 253–261.

Peylo, C., Thelen, T., Rollinger, C., & Gust, H. (2000). A Web-based intelligent educational system for PROLOG. In Peylo, C. (Ed.): *Proceedings of the International Workshop on Adaptive and Intelligent Web-based Educational Systems* (held in Conjunction with ITS 2000), Technical Report of the Institute for Semantic Information Processing, Osnabrück, (pp. 85–96).

Ritter, S. (1997). PAT Online: A model-tracing tutor on the world-wide web. In P. Brusilovsky, K. Nakabayashi, & S. Ritter (Eds.), *Proceedings of Workshop Intelligent Educational Systems on the World Wide Web at 8th World Conference on Artificial Intelligence in Education* (pp. 11–17). Kobe, Japan: ISIR.

Shute, V., & Glaser, R. (1990). A large-scale evaluation of an intelligent discovery world: Smithtown. *Interactive Learning Environments*, *1*(1), 51–77.

Thurman, D. A., Brann, D. M., & Mitchell, C. M. (1997). An architecture to support incremental automation of complex systems. *Proceedings of the 1997 IEEE International Conference on Systems, Man and Cybernetics*, 1174–1179.

Twidale, M. (1992). Knowledge acquisition for intelligent tutoring systems. In F. L. Engel, D. G. Bouwhuis, T. Bösser, & G. d'Ydewalle (Eds.), *Cognitive modelling and interactive environments in language learning, NATO ASI Series. SERS. F* (pp. Springer–Verlag). Berlin.

Virvou, M., & Moundridou, M. (2000). A Web-based authoring tool for Algebra-related ITSs. *Educational Technology & Society*, *3*(2), 61–70.

Walczak, S. (1998). Knowledge acquisition and knowledge representation with class: the object-oriented paradigm. *Expert Systems with Applications*, *15*, 235–244.

Webb, G. I., Pazzani, M. J., & Billsus, D. (2001). Machine learning for user modeling. *User Modeling and User-Adapted Interaction*, *11*(1–2), 19–29.

Woolf, B. P., & Cunningham, P. A. (1987). Multiple knowledge sources in intelligent teaching systems. *IEEE Expert*, *2*(2), 41–54.

Zukerman, I., & Albrecht, D. W. (2001). Predictive statistical models for user modeling. *User Modeling and User-Adapted Interaction*, *11*(1–2), 5–18.