Student and Instructor Models: Two Kinds of User Model and Their Interaction in an ITS Authoring Tool

Maria Virvou and Maria Moundridou

Department of Informatics, University of Piraeus,
80, Karaoli and Dimitriou St., Piraeus 185 34, Greece
{mvirvou, mariam}@unipi.gr

Abstract. WEAR is a Web-based authoring tool for Intelligent Tutoring Systems in Algebra related domains. Apart from modelling the student which is a common practice in almost all ITSs and ITS authoring tools, WEAR deals also with modelling the other class of its users: the instructors. Student and instructor models in WEAR interact with each other by exchanging information. This is in favour of both classes of WEAR's users, since they are affected by each other in a way similar to the one in a real educational setting. This paper describes the two kinds of user model and the type of information that they exchange. The issues raised in this research may be applied to other authoring tools by the addition of an instructor modelling component.

1 Introduction

One-to-one tutoring is believed (e.g. [1]) to be one of the most effective methods of instruction. Unfortunately, the large number of expert instructors that would be needed in such an educational setting make this ideal form of instruction unfeasible. Intelligent Tutoring Systems (ITSs) are computer-based instructional systems aiming at providing each student with a learning experience similar to the ideal one-to-one tutoring. In particular, ITSs have the ability to present the teaching material in a flexible way and to provide learners with tailored instruction and feedback. A number of successful evaluations of ITSs (e.g. [5; 9]) have managed to show that such systems can be effective in improving learning by increasing the students' motivation and performance in comparison with traditional instructional methods. However, ITSs are still seen with scepticism due to the fact that they have not been extensively used in real educational settings such as workplaces and classrooms. The main reason for this limited use is probably the fact that the task of constructing an ITS is complex, time-consuming and involves a large number of people including programmers, instructors and experts of a specific domain. Moreover, once constructed, an ITS for a specific domain can not be re-used for different domains without spending much time and effort. An approach to simplifying the ITS construction is to develop ITS authoring tools/shells. The main aim of such systems is to provide an environment that can be used by a wider range of people to easily develop cost-effective ITSs.

In the last decade a lot of research energy has been put in building ITS authoring tools; for a thorough and in-depth analysis of the state of the art for ITS authoring tools/shells the reader is referred to [6]. The users of ITS authoring tools are instructors who are responsible for the authoring procedure and learners who work with the produced ITSs. While learner modelling is a common task that is performed in almost every ITS and in many ITS authoring tools, instructor modelling has not gained any attention yet. This is an observation made also by Kinshuk and Patel [4]: "Whereas the work on student modelling has benefited by the user modelling research in the field of HCI, the research on the role of a teacher as a collaborator in the computer integrated learning environments is almost non existent." However, the role of instructors as users/authors of ITS authoring tools is very important for the effectiveness of the produced ITSs. In order for authoring tools to benefit the most from the involvement of instructors, they should provide individualised feedback to them throughout the ITS's life cycle. This can be achieved by an instructor modelling component incorporated in the architecture of the authoring tool.

Indeed, this paper is about an ITS authoring tool for the Web that models not only its students-users but also the instructors who author the ITSs to be generated. The authoring tool is called WEAR, which stands for WEb-based authoring tool for Algebra Related domains [10; 11]. WEAR provides a learning environment in which students can learn how to solve problems in various algebra-related domains (e.g. economics, physics, chemistry, etc.). In particular, WEAR deals with the generation of instruction, since it offers the ability of problem construction. In addition, it performs student error diagnosis by providing a mechanism that can be applied to many algebra-related domains. Finally, WEAR is also concerned with managing the sequence of the curriculum.

In particular, this paper focuses on the student and instructor modelling components of WEAR and their interaction. WEAR's user models (instructor and student model) interact with each other by exchanging information. This communication mimics in some sense the interaction that takes place in a real setting of a one-to-one tutoring: both the instructor and the student build models of each other and these models affect their attitude towards the learning process.

WEAR's design and development is based on the results of an empirical study that we conducted and which involved students and instructors of various algebra-related domains. What is of interest from the results of the empirical study in respect to the focus of this paper are the following: Instructors usually wish to know how students have performed so that they may evaluate their courseware with respect to their teaching goals. On the other hand, students always seek information about their instructor's teaching style, so that they may have a better understanding of what to expect from the course they are taught.

In the main body of this paper we will present WEAR's architecture and operation, describe how instructor and student modelling are incorporated in it, and also how and on what grounds these two models interact with each other.

2 Student and Instructor Models in WEAR's Architecture

WEAR is implemented in JAVA and PROLOG and resides on a Web server. Students and instructors can access WEAR and work with it using a conventional Web browser. The system's underlying architecture is shown in Figure 1. The "Tutoring components" consist of components that interact with students while they are solving problems, diagnose the cause of the errors a student may make, adaptively present the teaching material and form individualised advice for students. These components update the "Student model" at every interaction of a student with the system and use the information kept in that model to perform the tasks mentioned above. The "Tutoring components" use information that they obtain from the "Authoring components" which interact with the instructor. In particular they use the domain knowledge provided by the instructor in order to compare the students' solutions to the correct one.

The domain knowledge which is provided by the instructor to the "Authoring components" consists of the domain description in terms of variables, equations and units of measure, and all the information needed to construct a problem (known and unknown variables, level of difficulty, etc.). In addition, the instructor provides information concerning the structure of the course and the teaching material. The "Authoring components" are taking the input from the instructors and in return they assist them in constructing new problems, retrieving previously created ones, and delivering teaching material and problems to students. The "Authoring components" also provide information to the "Instructor model" which is updated at the end of each interaction of the instructor with the system. This model holds information acquired from the instructors in an explicit and implicit way. For example, instructors may be explicitly asked questions such as what their preferences are concerning the course, what their teaching expertise is, etc.; implicit information inferred by the system may concern the instructors' interests in some categories of problem etc. Using the information stored in the "Instructor model", the "Authoring components" render the system more flexible and adaptable to particular instructors' interests and needs.

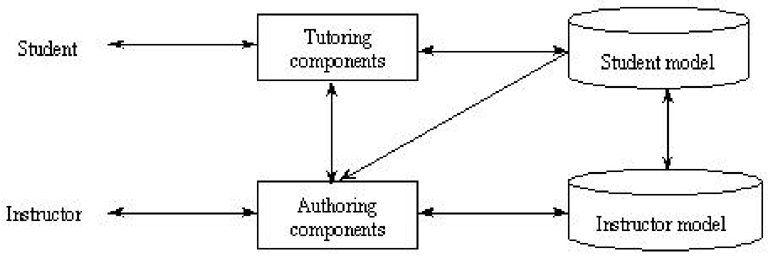

Fig. 1. WEAR's architecture

Most systems exploit the user models they build for the sake of the users to whom these models belong; for example a student model is used for the sake of students. However, WEAR uses its instructor and student model for instructors and students respectively and 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 in two ways: either explicitly by informing the other class of user or implicitly by affecting the model of the other class of user. As it can be seen from the system’s architecture (the arrow connecting the "Student model" with the "Authoring components"), the information that is explicitly provided, concerns the case of the student models, which pass information to instructors. 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.

Furthermore and 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 him/her of this when constructing 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.

WEAR functions in two different modes: the instructor's and the student's mode. The student's mode is in fact the ITS that WEAR produces, while the instructor's mode is the authoring tool itself. In the student's mode, students are presented with a number of problems to work with and are provided with individualised feedback while they are solving them. In the instructor's mode the instructor is able to construct new problems and/or retrieve previously created ones. In both cases, WEAR provides automatic assistance, as it will be discussed in the subsequent sections.

3 WEAR's Operation for the Instructor

The tool takes input from a human instructor about a specific equation-related domain (e.g. physics). This input consists of knowledge about variables, units of measure, formulae and their relation and can be provided 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 physics could be the following:

Variables: period - T, frequency - f, speed - u, wavelength - ë, time - t, distance - d

Equations: T=1/f, u=f*ë, U=d/t

Furthermore, instructors can upload HTML files and create a list of topics for the students to study. In that case, instructors are requested to provide some more information (e.g. they are asked to specify the prerequisite relationships among topics, associate the topics with the domain concepts, etc.). At this mode, instructors are also asked to answer some questions of the system concerning their preferences in teaching strategies and their level of expertise.

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. In particular, the problem construction procedure is the following: the system displays every variable that the human instructor has entered when describing the domain and requests the unknown. The system considers automatically all the variables as possible given data. Such variables depend on the "unknown" according to the domain equations. These variables are shown to the instructor who should now select the ones that will be "known" in the problem to be constructed and enter their values. 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. Finally, the system produces a simple problem text describing the given and asked data. After the construction of a problem, the tool asks the instructor to assign to the problem the appropriate "level of difficulty". While in student's mode the system uses this measure in order to suggest to each student what problem to try next.

Beyond constructing a problem by himself/herself, the instructor has the ability to explore the problems constructed by other instructors and choose the ones that s/he desires to be accessible by his/her class. Instructors are allowed either to browse the collection of problems by selecting the categories of problem that match their needs and interests, or to search the entire collection using some keywords. Problems are categorised according to the domain they belong. At a second level the problems of each domain are categorised according to the variables that they involve and their level of difficulty.

4 Instructor Modelling

The instructor modelling component monitors each instructor's interactions with WEAR and constructs and/or updates his/her user model. As we have already mentioned, an instructor either searches for an already constructed problem or constructs a problem by himself/herself. WEAR infers from these two actions the user's interest or contribution to something, respectively. This is similar to a system called InfoVine [3], which infers that users are interested in something if they are repeatedly asking the system about it whereas they are expert in something if they are repeatedly telling the system about it. In WEAR, when a user frequently searches for specific categories of problem then it is inferred that this particular user is "interested" in these categories of problem; when a user frequently constructs problems that belong to the same category and these problems have been solved by a number of students, the inference made is that this user is a "major contributor" in that sort of problem. In particular, the instructor aspects that are being modelled in WEAR, are the following:

Instructor's preferences: In WEAR the instructor may explicitly 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. The preference about the difficulty level of the course is associated with a percentage of failure in performances of class students. The preference about the level of popularity of the course is associated with the students' interest in the course (e.g. how many times students access the course, and/or the proportion of problems solved to the total number of available problems). In addition, the instructor may state if s/he considers equally important the two categories of students' error (mathematical and domain errors). In other words, the instructor is asked to specify how the students' level of knowledge will be calculated: if an instructor has stated that s/he considers more important the domain errors, then the weight of these errors will be higher in the formula calculating the students' level of knowledge.

Instructor's usual activities and special interests: Instructor's activities that are frequent are recorded in his/her long-term model. For example, if an instructor constructs exercises frequently then s/he is recorded as a major contributor. The instructor's interests are inferred and also recorded in the long-term instructor model. For example, what kind of exercises the instructor is interested in or whether s/he is interested in the diagnostic statistics about students depending on the number of times s/he visits the relevant page.

Instructor's level of expertise in teaching: 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, experienced. In the case of novice tutors and those having little experience, the authoring tool offers more detailed help concerning the teaching strategies that tutors may select and shows them the results of the consistency checks. In addition, WEAR after consulting the student models infers and records in instructor models the instructors' tendency to overestimate or underestimate the level of difficulty they assign to problems.

This user model is utilised by the system in the following ways:

Provide individualised help to the instructor. WEAR uses the instructor model in order to offer individualised help to the instructor with respect to his/her teaching strategies. 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 short term goals are consistent with his/her long-term goals and let him/her know accordingly. Student models provide information about how many students have attempted certain exercises and how many times they have seen certain lectures. This information is used to let the instructor know how well s/he does with his/her predefined teaching strategy. Similarly, if an instructor has been recorded in his/her long-term model as having the tendency to overestimate or underestimate the level of difficulty of problems, s/he will be reminded of that by the authoring tool when inserting new problems.

Adapt the interaction with its users. When a user 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. Of course, the instructor is free to select some other categories as well, or even ignore the already selected ones. In addition, if new problems belonging to the categories that a particular user is interested in are added, the system informs him/her when s/he logs in.

Promote co-operative or 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. The similarity of courses is calculated in terms of the domain to which they belong and in terms of the difficulty level assigned to them by their authors. In particular, the instructor may see an enriched Table of Contents presenting not only the topic hierarchy but also the relationships between topics. 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 users who are considered by the system as "major contributors" in the categories that this specific user is considered "interested". In addition, while an instructor is constructing a new problem by himself/herself the system is checking whether there is any similar problem already constructed by another instructor who is considered "major contributor". If this is the case, the instructor is offered the choice to see the similar problems and use them instead of completing the construction of his/her own. In that way, the system avoids the repetition of problems, facilitates the instructors' work and advances the co-operation and collaboration among them.

5 WEAR's Operation for the Student

Each student is assigned a level of knowledge by the system according to his/her past performance in solving problems with the tool. WEAR adaptively annotates the links to the problems that a student sees based on the student's "level of knowledge" and the "level of difficulty" that is assigned to each problem by the instructor. This is done in order to facilitate the student's choice about which problem to solve next resulting in adaptive navigation support [2]. In a similar manner WEAR provides adaptive navigation support to students concerning the topics that they should study.

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 such as: "The annoying sound from a mosquito is produced when it beats its wings at the average rate of 600 wingbeats per second. What is the frequency in Hertz of the sound wave? Assuming the sound wave moves with a velocity of 340 m/s, what is the wavelength of the wave?"[1]. 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 system's solution is generated by the domain knowledge about algebraic equations and about the specific domain where the problem belongs to (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 diagnostic component of WEAR attempts to diagnose the cause of it.

6 Student Modelling

A "history mechanism" embodied in WEAR records certain student features that have been inferred during past interactions, such as persistence of a certain type of error (e.g. mathematical error). These features form the long-term student model [7; 8] which represents the student's knowledge both in the domain being taught and in solving linear equations. This student model is a combination of a stereotype and an overlay student model. The stereotype student model (formed either directly by the instructor or after a preliminary test posed to the student) classifies initially the student according to his/her knowledge of the domain and his/her mathematical skills. As a result of this, the student is assigned to a stereotype (novice, beginner, intermediate, or expert). For example, a student may be assigned to the stereotype "expert" for his/her mathematical skills and to "beginner" for his/her knowledge of the

[1] This example problem statement is taken from:
http://www.glenbrook.k12.il.us/gbssci/phys/Class/waves/u10l2e.html

domain taught. The stereotype model also defines initial values for the overlay student model. The latter is represented by a set of pairs "concept-value" which are explained below.

The concepts are domain concepts and concepts concerning the process of solving equations (e.g. separating known from the unknown). Domain concepts include domain variables. For example, in the domain of Physics the variables "Velocity", "Force", etc. are seen as concepts. It is assumed by the system that a student knows a concept if in a given problem that this variable-concept is needed, s/he enters the correct equation that defines this variable. The value for each concept is an estimation of the student's knowledge level of this concept.

The student model is used by the system in various ways. There are cases when an error that a student makes can be attributed to more than one cause. In such cases the student model that holds information concerning the student’s tendency to particular types of error is used to resolve the ambiguity as to what the cause of the error has been. In addition, the student model is used to form individualised progress reports of the student, which could be requested by the student and/or the instructor, as well as to provide adaptive navigation support to students concerning which problem to solve next and which topic to study. Furthermore, student models are used by the system to inform the instructor about problematic situations (e.g. when the majority of students fail to comprehend something, as may be indicated by their low scores in the corresponding tests). Finally, student models provide evidence as to whether the instructor's long-term goals (popularity and/or difficulty of the course) are far from being achieved or not.

7 Interaction between Student and Instructor Models

As already discussed in the above sections, instructor and student models acquire information about instructors and students respectively, in an explicit and implicit manner. An important source for the implicit acquisition of information for the model of each class is the model of the other class of user. This is done so that each class of user may receive help and feedback taking into account the relevant issues that are determined by the other class.

In the case of instructor modelling, student models provide information about how many problems the students have attempted to solve and how many times they have visited certain topics of the teaching material. This information is used to let the instructor know how well s/he does with his/her predefined goal of popularity of the course.

Another issue that concerns the instructor model comes up while students are tackling the given problems. In this case, the system collects evidence about the level of difficulty of these problems 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 by the instructor 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.

Finally, instructor models hold information about the instructors' contribution to particular categories of problem. For an instructor to be considered a "major contributor" in a specific area, s/he should have created many problems in that area and these problems must have been solved by a number of students. This information is again provided by the student models.

In the case of student modelling, the students' level of knowledge is affected both by the domain and mathematical errors that they may make. The students' level of knowledge is calculated taking into account the instructor's preference to assign more weight to one or the other category of error. In this way the instructor model affects the student modelling procedure.

Another piece of information that is implicitly passed from the instructor model to the student model concerns the level of difficulty of the course. In particular, students have the choice of seeing what their level of knowledge is, not only in absolute terms but also in relation to the level of difficulty of the course, as stated by the instructor when defining his/her long-term goals. For example, a student's performance in a course may be average instead of very good due to the fact that the instructor's goal was to create a quite difficult course. Having the ability to see his/her real knowledge level and also a knowledge level affected in some way by the course's level of difficulty, the student can better realise where s/he stands in the domain being taught.

8 Conclusions

In this paper we presented WEAR, a Web-based authoring tool for Intelligent Tutoring Systems in Algebra related domains, and focussed on its user modelling capabilities. In WEAR we model two classes of user: the students and the instructors - unlike most ITS authoring tools that only model the students. We argued that an instructor modelling component may render the system more flexible and adaptable to particular instructors' interests and needs. Moreover, we discussed an issue that we consider important: the interaction between the instructor and student models. Information stored in student models is passed either to the instructor himself/herself or to his/her user model. The same happens with instructor models that pass information to students and student models. What motivated us to implement this information exchange between user models, was the observation that in a real one-to-one tutoring setting, both students and instructors build models of each other and based on that models they adjust the way they respond during the learning process.

From an empirical study we conducted involving both students and instructors, there is strong evidence that the user modelling mechanisms that WEAR embodies may be really in favour of both classes of user. However, in the near future we plan to evaluate WEAR in whole and especially the user modelling aspects discussed in this paper.

References

1. Bloom, B.: The 2 sigma problem: The search for methods of instruction as effective as one-to-one tutoring. *Educational Researcher*. 13(6) (1984) 4-16
2. Brusilovsky, P.: Methods and techniques of adaptive hypermedia. *User Modeling and User-Adapted Interaction*. 6 (2-3) (1996) 87-129
3. Harvey, C.F., Smith, P. & Lund, P.: Providing a networked future for interpersonal information retrieval: InfoVine and user modelling. *Interacting with Computers*. 10 (1998) 195-212
4. Kinshuk & Patel, A.: Intelligent Tutoring Tools: Redesigning ITSs for Adequate Knowledge Transfer Emphasis. In Lucas, C. (ed.): *Proceedings of 1996 International Conference on Intelligent and Cognitive Systems*. IPM, Tehran (1996) 221-226
5. Koedinger, K.R. & Anderson, J.R.: Intelligent Tutoring Goes to School in the Big City. *International Journal of Artificial Intelligence in Education*. 8 (1997) 30-43
6. Murray, T.: Authoring Intelligent Tutoring Systems: An analysis of the state of the art. *International Journal of Artificial Intelligence in Education*. 10 (1999) 98-129
7. Rich, E.: User Modelling via Stereotypes. *Cognitive Science*. 3 (4) (1979) 329-354
8. Rich, E.: Users as Individuals: Individualizing User Models. *International Journal of Man-Machine Studies*. 18 (1983) 199-214
9. Shute, V., Glaser, R. & Raghaven, K.: Inference and Discovery in an Exploratory Laboratory. In Ackerman, P.L., Sternberg, R.J. & Glaser, R. (eds.): *Learning and Individual Differences*. Freeman, San Francisco (1989) 279-326
10. Virvou, M. & Moundridou, M.: A Web-Based Authoring Tool for Algebra-Related Intelligent Tutoring Systems. *Educational Technology & Society*. 3(2) (2000) 61-70
11. Virvou, M. & Moundridou, M.: Modelling the instructor in a Web-based authoring tool for Algebra-related ITSs. In Gauthier, G., Frasson, C. & VanLehn, K. (eds.): *Intelligent Tutoring Systems, Proceedings of the 5th International Conference on Intelligent Tutoring Systems, ITS 2000*. Lecture Notes in Computer Science, Vol. 1839. Springer, Berlin (2000) 635-644