2nd Hellenic Conf. on AI, SETN-2002, 11-12 April 2002, Thessaloniki, Greece,
Proceedings, Companion Volume, pp. 203-214

WEAR: A Web-Based Authoring Tool for Building Intelligent Tutoring Systems

Maria Moundridou & Maria Virvou

Department of Informatics, University of Piraeus
80 Karaoli & Dimitriou St., Piraeus 18534, Greece
Phones: +3014142133, +3014142269
Fax: +3014142264
{mariam, mvirvou}@unipi.gr

Abstract. WEAR is a Web-based authoring tool for the construction of Intelligent Tutoring Systems (ITSs) in Algebra-related domains, such as physics, economics, chemistry, etc. In WEAR's authoring environment instructors are able to construct problems and tests and also build adaptive electronic textbooks. In return, WEAR generates a learning environment in which students can solve problems and study the topics of the curriculum. 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. Based on the user models it maintains, WEAR adapts the interaction with both students and instructors and provides them with individualised feedback and help. In this paper we will describe WEAR's operation and functionality and discuss how this operation is enhanced by the system's user modelling capabilities.

1 Introduction

One-to-one tutoring is believed to be one of the most effective methods of instruction (e.g. [1]). 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], [11]) 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. 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 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. Murray in his paper reviewing the state of the art for ITS authoring tools [10], has classified these systems based on their capabilities and concluded that they fall into two broad categories: those which focus on how to sequence and teach relatively canned content (pedagogy-oriented authoring tools) and those which focus on providing rich learning environments in which students can learn skills by practicing them and receiving feedback (performance-oriented authoring tools).

WEAR, the system we will describe in this paper, mainly belongs to the category of performance-oriented authoring tools, since it provides a learning environment in which students can learn how to solve problems in various algebra-related domains. In particular, WEAR deals with the generation of instruction, since it offers the ability of problem construction and also the ability of building adaptive electronic textbooks. In that sense it shares the same focus with RIDES [9], an authoring system used for the construction of tutors that teach students how to operate devices through simulations. A system which adds capabilities to RIDES is DIAG [12], a tool which simulates equipment faults and guides students through their diagnosis and repair. DIAG is concerned with the creation of domain knowledge and performs student error diagnosis by providing a mechanism that is applicable to many domains that are related to diagnosis of equipment failures. In the same way WEAR performs student error diagnosis by providing a mechanism that can be applied to many algebra-related domains.

However, WEAR also shares capabilities with authoring tools belonging to the pedagogy-oriented category. In particular, WEAR gives instructors the ability to control the order by which students solve problems and study the teaching material by assigning a value to each problem's attribute called "level of difficulty" and by defining prerequisite relationships between topics of the electronic textbook. Therefore, WEAR is also concerned with managing the sequence of the curriculum on top of generating it. The former is a characteristic that can likewise be met in a system called REDEEM (Major, Ainsworth & Wood, 1997). REDEEM expects the human instructor to describe existing teaching material in terms of their difficulty, their generality, etc., to construct teaching strategies (i.e. when and how to test the students, how much hinting and feedback to offer, etc.) and to identify students. The tool exploits the knowledge provided by the instructor and its default teaching knowledge to deliver individualised instruction to students. Another pedagogy-oriented system is GTE [13]. GTE allows an author to develop courseware declaratively, through the creation of various instructional objects (exercises, presentations, examples, etc.) that make up a course. The central component of GTE's architecture is a large generic instructional knowledge base containing instructional tasks and methods. This knowledge base makes GTE able to take the declarative courseware specification that an author has given and use it in a real instructional context.

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 in [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 [17].

Indeed, WEAR is 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 [15]. Furthermore, WEAR's user models (instructor and student model) interact with each other by exchanging information [16]. 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.

This paper is a review of the current version of WEAR which incorporates new features, such as the adaptive navigation support provided to students and the instructor modelling mechanisms that are used for offering intelligent and tailored assistance to instructors. An earlier version of WEAR, focusing on problem construction and solving, was described in [14]. In the main body of this paper we will present WEAR's architecture and operation and describe how the student and instructor modelling components are incorporated in it.

2 Wear's Architecture

The system's underlying architecture is shown in Figure 1. 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 instructor passes to the Authoring components forms the database of Domain knowledge and problems.

The Instructor modeller is responsible for building and updating each instructor model. The Instructor model holds: i) information obtained explicitly by asking the instructors (such information may be the instructor's preferences concerning the course and his/her teaching expertise), and ii) 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 s/he finds already pre-selected those categories that s/he is interested in.

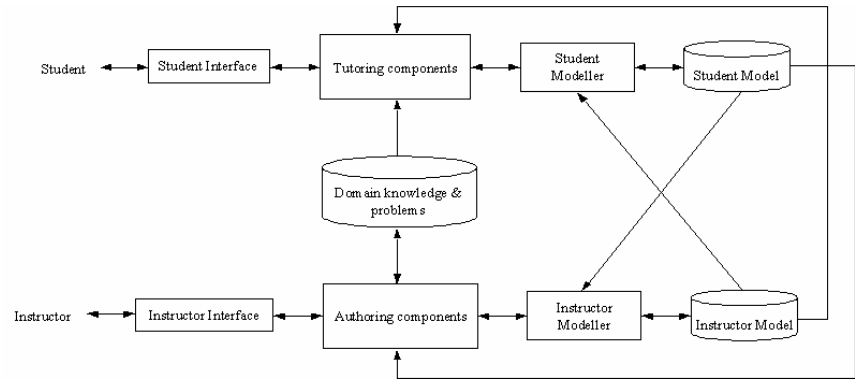

Fig. 1. WEAR's architecture

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, what the structure of the domain being taught is (e.g. which are the prerequisite concepts a student should know before studying a specific concept) and which the correct equations that describe this domain are. The sources for all this information are the Student models and the Domain knowledge and Problems. A Problem Solver included in the Tutoring components is using its knowledge about solving systems of linear equations correctly in order to inform the Student modeller about the problem solving activity of the student. The Problem Solver is also using information from the Domain knowledge and problems. In case of an error, the Student modeller is responsible for diagnosing the cause of it. The Student modeller is also responsible for updating the Student model based on the student's actions when interacting with the system (reading or not topics of the electronic textbook, solving correctly or not a problem, etc.).

The resulting ITSs from WEAR have a Student interface that incorporates a speech-driven animated agent which provides speaking messages to the student [8]. On the other hand, the Instructor interface does not include any animated agent and it operates as a conventional GUI.

As shown in the above figure, 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, 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 the instructor (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.

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 WEAR's Operation

WEAR functions in two different modes: the instructor's mode and the student's mode. The instructor's mode provides the environment of the authoring tool itself while the student's mode provides the environment for the ITS that WEAR produces. In the student's mode, 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 instructor's mode the instructor is able to construct new problems, retrieve previously created ones and author the adaptive electronic textbook. In all cases, WEAR provides automatic assistance, as will be discussed in the subsequent sections.

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. 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.

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

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 procedure of constructing a problem 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, which

depend on the "unknown" (according to the equations), as possible given data. These variables are shown to the instructor who should now 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, which the instructor may change 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 (Figure 2). After the construction of a problem the tool lets the instructor preview the problem text and the solution of the problem as formulated by the system. At this point, the instructor is asked to assign to the problem the appropriate level of difficulty. The system uses this measure in order to suggest to each student what problem to try next.

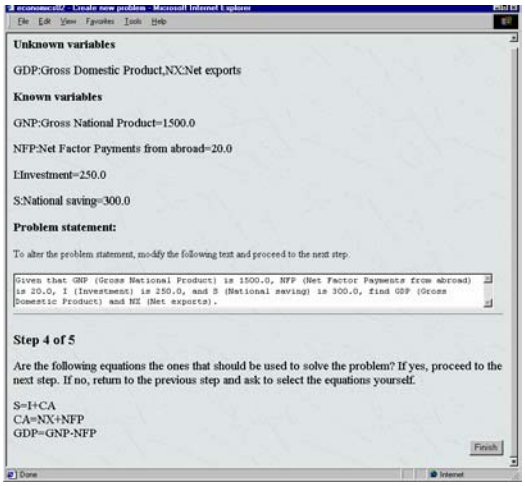

Fig. 2. Problem construction

While students are tackling 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 this kind of problem in which the students are tested over their ability to solve a

system of linear equations (mathematical skills) and their knowledge of the equations describing the particular domain, WEAR also offers instructors the ability to create multiple-choice tests. Since there are topics of the curriculum that can not be assessed by problems such as the above mentioned, by using multiple-choice tests instructors can be aware of their students' performance and understanding in these topics as well.

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. Since new problems (belonging to different domains, involving different variables, etc.) can be continuously added to the system, there is no way for the system to have fixed categories of problem. 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. An instructor modelling mechanism incorporated in the system is responsible for tailoring the interaction of the instructors with the system to the instructors' needs. The exact way, in which this adaptation of the interaction to the instructors' needs is performed, is described in section "Instructor modelling".

Finally, WEAR allows the authoring of electronic textbooks by instructors and delivers them over the WWW to learners [7]. 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 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. Finally, the instructor must edit the *is_prerequisite_of* and *is_related_to* relationships between topics. This information is used to form WEAR's domain model. The domain and student models are used by WEAR to generate a table of contents for each student. This table of contents consists of links to each topic of the textbook. These links are annotated in order to inform students about the educational appropriateness of the topic behind them. 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.

3.2 The Learning Environment

Information obtained from student models as well as knowledge of the domain being taught, are exploited by WEAR to provide adaptive navigation support to students [2]. 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.

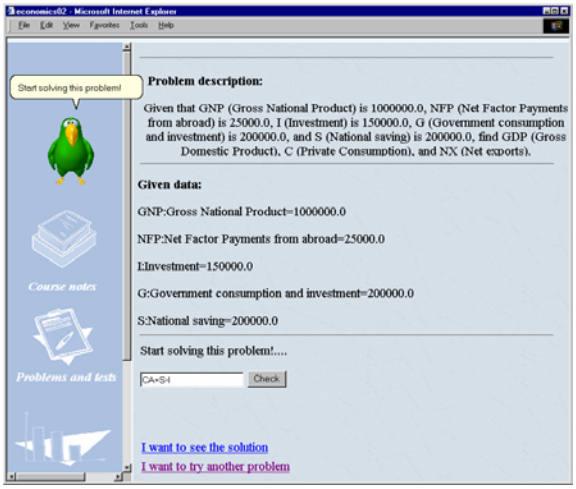

Fig. 3. Solving a problem while in student's mode

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 Figure 3. 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 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. Based on this diagnosis, the system provides the student with the appropriate feedback message. As has been already mentioned, the student interface includes an animated speaking character which is responsible for communicating the instructions and any feedback messages to the students.

4 User Modelling in WEAR

4.1 Instructor Modelling

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 a percentage of failure in performances of class students and student interest in the course (e.g. 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 short-term goals are consistent with his/her long-term goals. 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".

4.2 Student Modelling

The student model that WEAR maintains is a combination of a stereotype and an overlay student model, similarly with other systems such as [3]. The stereotype student model (formed either directly by the instructor or after a preliminary test that has been 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, 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 "intermediate" 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 the corresponding values: 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.

As has been already mentioned, during the process of solving a problem the student's actions are monitored by the system and 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 the following:

1. *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 U=d*t instead of U=d/t, then the error is attributed to the category of Domain errors and in particular to the sub-category of "erroneous relationship between variables".
2. *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 d in the equation U=d/t enters d=U/t instead of d=U*t, 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".

5 Conclusions

In this paper we described WEAR which is a Web-based authoring tool for the construction of Intelligent Tutoring Systems in Algebra-related domains. In WEAR's authoring environment instructors are able to construct problems and tests and also build adaptive electronic textbooks. In return, WEAR generates a learning environment in which students can solve problems and study the topics of the curriculum. An important aspect of the system is its user modelling capabilities. In WEAR both classes of user (students and instructors) are being modelled, unlike what is happening with most ITS authoring tools that only model the students. Based on the user models it maintains, WEAR can tailor the interaction with each user. In the near future we plan to evaluate WEAR in whole and especially the user modelling issues 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. Hohl, H., Böcker, H., Gunzenhäuser, R.: Hypadapter: An Adaptive Hypertext System for Exploratory Learning and Programming. User Modeling and User Adapted Interaction, 6(2-3) (1996) 131-155
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., Hadley, W., Mark, M.: Intelligent Tutoring Goes to School in the Big City. International Journal of Artificial Intelligence in Education. 8 (1997) 30-43
6. Major, N., Ainsworth, S., Wood, D.: REDEEM: Exploiting Symbiosis Between Psychology and Authoring Environments. International Journal of Artificial Intelligence in Education. 8 (1997) 317-340
7. Moundridou, M., Virvou, M.: Authoring and Delivering Adaptive Web-Based Textbooks using WEAR. In: Okamoto, T., Hartley, R., Kinshuk, Klus, J.P. (eds.): IEEE International

Conference on Advanced Learning Technologies; Issues, Achievements, and Challenges - ICALT 2001. IEEE Computer Society, Los Alamitos California (2001) 185-188
8. Moundridou, M., Virvou, M.: Evaluating the Impact of Interface Agents in an Intelligent Tutoring Systems Authoring Tool. In: Avouris, N., Fakotakis, N. (eds.): Advances in Human-Computer Interaction I: Proceedings of the Panhellenic Conference with International participation in Human-Computer interaction – PC-HCI 2001. Typorama Publications, Patras Greece (2001) 371-376
9. Munro, A., Johnson, M., Pizzini, Q., Surmon, D., Towne, D., Wogulis, J.: Authoring Simulation-centered tutors with RIDES. International Journal of Artificial Intelligence in Education. 8 (1997) 284-316
10. 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
11. Shute, V., Glaser, R.: A large-scale evaluation of an intelligent discovery world: Smithtown. Interactive Learning Environments. 1(1) (1990) 51-77
12. Towne, D.: Approximate reasoning techniques for intelligent diagnostic instruction. International Journal of Artificial Intelligence in Education. 8 (1997) 262-283
13. Van Marcke, K.: GTE: An epistemological approach to instructional modelling. Instructional Science. 26 (1998) 147-191
14. Virvou, M., Moundridou, M.: A Web-Based Authoring Tool for Algebra-Related Intelligent Tutoring Systems. Educational Technology & Society. 3(2) (2000) 61-70
15. Virvou, M., Moundridou, M.: Modelling the instructor in a Web-based authoring tool for Algebra-related ITSs. In: Gauthier, G., Frasson, C. and VanLehn, K. (eds.): Intelligent Tutoring Systems: Proceedings of the 5th International Conference - ITS 2000. Lecture Notes in Computer Science, 1839. Springer, Berlin (2000) 635-644
16. Virvou, M., Moundridou, M.: Student and Instructor Models: Two Kinds of User Model and their Interaction in an ITS Authoring Tool. In: Bauer, M., Gmytrasiewicz, P., Vassileva, J. (eds.): User Modeling 2001: Proceedings of the 8th International Conference UM2001. Lecture Notes in Artificial Intelligence, 2109. Springer, Berlin (2001) 158-167
17. Virvou, M., Moundridou, M.: Adding an Instructor Modelling Component to the Architecture of ITS Authoring Tools. International Journal of Artificial Intelligence in Education. 12 (2001) 185-211