# Modern Educational Tools

***Boban Vesin[1], Mirjana Ivanović[2]***

*[1]Bussines School, Novi Sad*
*e-mail: vesinboban@yahoo.com*

*[2]Department for Mathematics and Informatics*
*Faculty of Science, Novi Sad*
*e-mail: mira@im.ns.ac.yu*

**Abstract:** The concept, known as intelligent tutoring systems has been pursued for more than 30 years by researchers in psychology, education, and artificial intelligence. Many of the intelligent tutoring systems that have been developed during that last three decades have proven to be quite successful, particularly in the domains of mathematics, science, and technology.

The goal of intelligent tutoring systems is to provide the benefits of one-on-one instruction automatically and cost effectively. Like training simulations, intelligent tutoring systems enable participants to practice their skills by carrying out tasks within highly interactive learning environments. However, intelligent tutoring systems go beyond training simulations by answering user questions and providing individualized guidance. Unlike other computer-based training technologies, intelligent tutoring systems assess each learner's actions within these interactive environments and develop a model of their knowledge, skills, and expertise. Based on the learner model, intelligent tutoring systems form instructional strategies, and provide explanations, hints, examples, demonstrations, and practice problems as needed. They have been shown to be highly effective at increasing student's performance and motivation.

In this paper, an overview of the main components of intelligent tutoring systems and different types of intelligent tutoring systems will be presented.

## 1. Introduction

Computers have been used in education for more than 30 years. In these kinds of systems, the instruction was not individualized to the student's needs. Instead, the decisions about how to move a student through the material were script-like, such as "if question 21 is answered correctly, proceed to question 54; otherwise go to question 32." The learner's abilities were not taken into account.

While first such systems may be somewhat effective in helping learners, they do not provide the same kind of individualized attention that a student would receive from a human tutor. For a computer based educational system to provide such attention, it must reason about the domain and the learner. This has caused research in the field of intelligent tutoring systems (ITSs). ITSs offer considerable flexibility in presentation of material and a greater ability to respond to individual student needs. These systems achieve their "intelligence" by representing pedagogical decisions about how to teach as well as information about the learner. This allows for greater versatility by altering the system's interactions with the student.

Intelligent tutoring systems have been shown to be highly effective at increasing students' performance and motivation.

This article briefly present components of Intelligent Tutoring Systems (chapter two), types of Intelligent Tutoring Systems (chapter three) and main elements of Student model (chapter four). Chapter five illustrates principles of two Intelligent Tutoring Systems: *Minerva 5.1* and *Pepite*.

## 2. Components of Intelligent Tutoring Systems

Intelligent tutoring systems may look to be monolithic systems, but for the purposes of conceptualization and design, it is often easier to think about them as consisting of several interdependent components. Previous research has identified five major components: the student model, the pedagogical module, the domain knowledge module, the expert model and the communication module [2]. Figure 1 provides a view of the interactions between the modules.

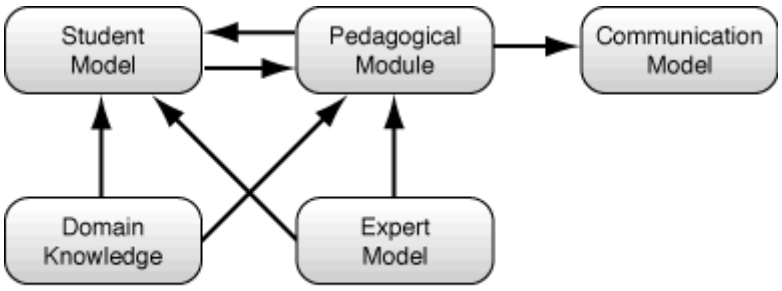



Figure 1: Interactions of components in an intelligent tutoring system.

### 2.1 Student Model

The student model stores information that is specific to each individual learner. At a minimum, such a model tracks how well a student is performing on the material being taught. A possible addition to this is to also record misconceptions. Since the purpose of the student model is to provide data for the pedagogical module of the system, all of the information gathered should be able to be used by the tutor.

### 2.2 Pedagogical Module

This component provides a model of the teaching process. For example, pedagogical module control giving information about when to review, when to present a new topic, and which topic to present. As mentioned earlier, the student model is used as input to this component, so the pedagogical decisions reflect the differing needs of each student.

### 2.3 Domain Knowledge

This component contains information what the tutor is teaching about. It is the most important part of the system since without it, there would be nothing to teach the student. Generally, it requires significant knowledge engineering to represent a domain so that other parts of the tutor can access it. One related research issue is how to represent knowledge so that it easily scales up to larger and different domains. Another open question is how to represent domain knowledge other than facts and procedures, such as concepts and mental models.

### 2.4 Communications Module

Interactions with the learner, including the dialogue and the screen layouts, are controlled by this component. How should the material be presented to the student in the most effective way? This

component in earlier systems has not been subject of the intensive research as other components. Nowdays, there has been some promising investigation and results in this area.

### 2.5 Expert Model

The expert model is similar to the domain. However, it is more than just a representation of the data; it is a model of how someone skilled in a particular domain represents the knowledge. Most commonly, this takes the form of a runnable expert model, i.e. one that is capable of solving problems in the domain. By using an expert model, the tutor can compare the learner's solution to the expert's solution, pinpointing the subject in which the learner had difficulties.

## 3. Types of ITSs

There are several ways of categorizing ITSs. We will concentrate on two dimensions: abstraction of the learning environment and the knowledge type of the instruction.

### 3.1 Abstraction of the learning environment

Many systems attempt to provide instruction by simulating a real environment in which the student can learn the task. There are many reasons for developing such systems, including the possible danger of training using the actual equipment and the lack of domain experts who can devote their expensive time to training novices. Therefore, a realistic simulated learning environment can reduce both the cost and the risks of training.

An example of a simulation-based ITS is the *ATC-TutorHelp***.** The domain of this ITS is Air Traffic Control Training. System is traying to teach student how to control air traffic by simulating a real situation in the control tower.

.

At the extreme opposite of the simulation based tutors are those that teach knowledge in a decontextualized manner without attempting to simulate the real situation. Many systems developed during the history of ITS research fall into this category. These systems provide problems to be solved without trying to connect those problems to a real world situation. They are designed to teach abstract knowledge that can be transferred to multiple problem solving situations.

### 3.2 Emphasis of Instruction

There is a long history of classifying instructional goals according to the type of knowledge being taught. An important early attempt at this classification is Bloom's taxonomy [3] and much recent work in categorizing knowledge has been derived from this. In addition to classifying learning goals by knowledge type, one can also examine what the student will be able to do upon completion of the ITS's lesson.

For the sake of development, systems tend to concentrate on teaching one type of knowledge. The most common type of ITS is that which teaches procedural skills. The goal is that students learn how to perform a particular task. There has been substantial research in cognitive psychology concerning human skill acquisition. Systems designed according to these principles are often called *cognitive tutors* [1]. The most common result of this analysis is a set of rules that are part of a runnable expert model. This set of expert rules often serves double duty as knowledge of the domain and as the pedagogical module. If a student encounters difficulty, the specific remediation required can be determined from the expert model.

An example of a ``cognitive tutor" is *Pepite*[6] which has tutorial actions associated with each state in the ``effective problem space" [4]. Another example of an ITS that uses an analysis of expert behavior is the *SQL-tutor*, which encodes expert problem solvers' actions as production rules, and attempts to determine which rules cause difficult for the student.

Other ITSs concentrate on teaching concepts and ``mental models" to students. These systems encounter two main difficulties. First, more substantial domain knowledge is needed for instruction. Second, since learning concepts and frameworks are less well understood than learning procedures, there is less cognitive theory to guide knowledge representation and the pedagogical module. For these reasons, ITSs of this type require a larger domain knowledge base and are sometimes referred to as *knowledge based tutors*. As a result of lack of a strong model of skill acquisition or expert performance, these systems are forced to use general teaching strategies. They also place more emphasis on the communication and presentation part of the system in order to achieve learning gains. An example of such a system is the *Minerva* [7].

These classifications are really points along a continuum, and serve as good rules of thumb rather than a definitive method of classifying intelligent tutoring systems. A system that does not fall into either of these categories is *Coach* [5], which teaches how to use UNIX mail. This is a procedural skill, and hence cognitive in nature.

Generally, tutors that teach procedural skills use a cognitive task analysis of expert behavior, while tutors that teach concepts and frameworks use a larger knowledge base and place more emphasis on communication to be effective in giving instruction. There are exceptions to these rules, but they serve as useful guidelines for classifying ITSs.

## 4. The Student Model

As mentioned in section 2.1, the student model is the component of an ITS that records information about the student. This information reflects the system's belief of the learner's current knowledge state. Since only student keyboard-actions are visible, and the ITS only has a relatively narrow channel of communication, there is difficulty in obtaining an accurate representation of the student's abilities. Therefore, the model of the student may not be perfectly accurate. Also, steps must be taken to ensure that the system's actions on the basis of this inaccurate information are not inappropriate. For example, a tutor that interferes too much with a learner who is performing correctly can obviously be detrimental.

After considering the above difficulties, an obvious question concerning student models is why to have one. Simply put, the student model is necessary in order to tailor instruction to a student's idiosyncrasies and learning needs. Without this knowledge, the pedagogical component has no basis for making decisions, and is forced to treat all students similarly.

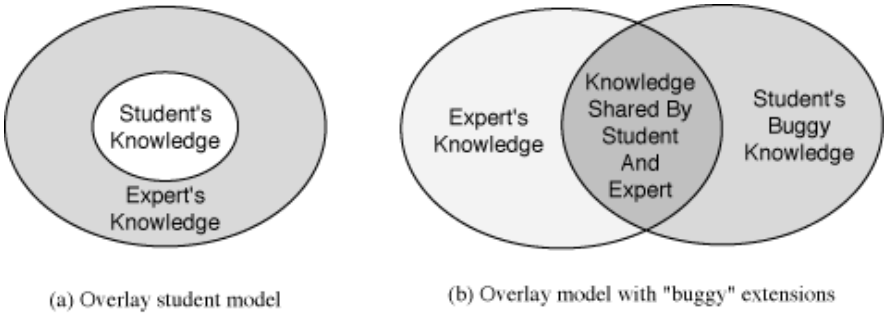
### 4.1 Representation of the student model

There are many methods for representing information about the student. Two commonly used techniques are overlay models and Bayesian networks.

The standard paradigm for representing a student model is the *overlay model* [3] in which the student's knowledge is considered to be a subset of the expert's knowledge (Figure 2a). In this case, an ITS presents material to the student, so that his knowledge will exactly match the expert’s one.

A drawback of this approach is that it does not acknowledge that students may have beliefs that are not part of the expert's knowledge base. For example, students frequently have misconceptions about a

domain. Therefore an extension to the overlay model explicitly represents “buggy” knowledge that the student may have (Figure 2b).




(a) Overlay student model (b) Overlay model with "buggy" extensions

Figure 2: Overlay student models.

Another mechanism for representing a student's knowledge is *Bayesian networks* [3]. These networks probabilistically reason about a student's knowledge state based on his interactions with the tutor. Each node in the network has a probability of the student “knowing” that piece of knowledge.

### 4.2 Content of the student model

ITS designers have a tendency to include more information in the student model than the pedagogical module can use. However, the point of having a student model is to be able to tailor the instruction for each student. Therefore, student models should only include data that the tutor will actually use in creating this instruction. On the other hand, for research purposes, it can be beneficial to include extra factors that are not used by the pedagogical module but provide ITS designers with knowledge of what may be useful to include in future student models.

Given this restriction, what should a student model contain? Clearly, it must include the student's understanding of the domain. However, at what degree should that understanding be represented? At one extreme, the tutor could simply say ``the student knows this domain" or ``the student does not know this domain." At the other extreme, the student model could record every student action. Most student models fall in between these two categories and try to model the student at the same granularity at which the domain is represented. Since most domains are represented in terms of topics, this is the most common grain size for student models.

In addition to record a student's understanding of the domain, a student model may also include more general pedagogical information about the student. This kind of information could include a student's general preferences, such as whether he likes to look at examples before attempting to answer some questions.

Other general information about the student's learning can be included, such as acquisition and retention [5]. *Acquisition* measures how fast students learn new topics, and *retention* measures how well they recall the material over time. Prior research suggests that examining general factors such as acquisition and retention can be beneficial for student modeling.

## 5. Software examples

### 5.1 *Pépite*

*Pépite* is free software available for download [6]. The first version was implemented in Delphi by S. Jean. Second version improved the automatic diagnosis, usability and effectiveness and has been implemented in Java. *Pépite* software consists of three modules.

*PepiTest* is the students' software. It proposes 22 exercises derived from the paper and pencil tasks and it gathers students' answers to problems. It contains closed questions with Multiple Choices answers or more interactive answering techniques (for instance matching of clickable parts of a graphic, deciding between limited number of possible answers). It contains also open questions requiring the student to produce algebraic expressions or natural language answers or mixed answers. For methodological purpose it is important that the students can provide explanations. A great attention has been paid to HCI design issues. It is crucial for the diagnostic to be sure that student's answers are regularly read. In particular, the mathematicians that are from project team were very suspicious at the beginning about the modifications in the mathematical tasks due to the difficulties to enter algebraic expressions using keyboard and mouse.

*PepiDiag* is the analysis module. It "interprets" students' answers to given. Like in the paper and pencil tool, it matches every student's answer with an item of diagnostic. We call that operation coding student's answers. *PepiDiag* automatically fills a "diagnostic matrix" of 55 questions and 35 items derived from the multidimensional model of algebraic competencies. In fact it partially fills the matrix. The closed answers and algebraic expressions are analyzed. Natural language answers and mixed answers are very partially analyzed by key words analysis. So 75 % of the students' answers are automatically analyzed.

*PepiProf* is software devoted to teachers. It establishes the student's profile from the filled matrix by transversal analysis and presents it to the teacher. It also provides an interface to modify student's answers (i.e. to modify the diagnosis matrix without showing it to the teacher) in order to allow the teacher to control the software coding and to correct or complete it when necessary.

### 5.2 *Minerva 5.1*

*Minerva* 5.1 is an e-learning tool intended to help in the study of any topic when it's necessary to memorize any kind of information. Minerva lets the teacher to create "study books" with "cards" (also called records, organized into subjects), each card with a question and an answer (and other related data), which will be asked to the student according to some criteria controlled by the teacher. The first version of *Minerva* was developed in 1999 and, since then, the program has evolved around four fundamental ideas, basis for learning:

- **It uses previously known stimulus**, like the images associated with every question, that makes it easier to memorize.
- **Re-learning**: the already known material is easier to memorize the second time it's studied. Minerva not only asks each record (each question) two times, but it follows the evolution of every record, it repeats a question as often as necessary so as to be sure that it has been correctly memorized.
- **The student is involved** in the process of learning. *Minerva* is showing him/her data (and charts) so as to see the evolution of the study: age of the books, right answers percentage, etc.

*Minerva's* objective is to help the student memorize all the cards, adapting its behavior at all times to the student's evolution, taking maximum advantage of his/her time.

*Minerva 5.2* (recent version) offers, besides the classical single-user mode, multi-user options, both in local and network mode. In multi-user mode, the network administrator (teacher, etc.) can use some personal and global analysis tools to manage students and books. The interface is translated into Spanish, English and Portuguese.

## 6. Conclusion

Intelligent tutoring systems have been seen as highly effective tool in increasing student motivation and learning. Usually they are implemented as a composition of five components: the student model,

the pedagogical module, the domain knowledge, the communications module, and the expert model. Research has been done on each of these modules, but only a few are very well understood. Specifically, incorporating multiple teaching strategies in the pedagogical module is a challenging open research question.

In addition to the continuing work on these components, one important research issue is reducing the time and cost to develop such systems. Current strategies include the development of authoring tools and creating systems in a modular fashion. Solving this problem will be an enormous step forward in ITS research.

References


[1] Lajoie, S.P. 1997. *Computers as Cognitive Tools*, Lawrence Erlbaum Associates, NJ

[2] Murray, T. 1996. *Intelligent Tutoring Systems architecture.* Iz *Proceedings of the Third International Conference on Intelligent Tutoring Systems*, Montreal

[3] Winkels, R., J. Sandberg, and J. Breuker. 1992. *Developing Intelligent Help Systems*, Breuker, J., ed., EC, Copenhagen

[4] Stern, M., J. Beck, and B. Woolf. 2003. *Intelligent Tutoring Systems: Lessons Learned*. Iz *Proceedings of the Third International Conference on Intelligent Tutoring Systems*, Montreal.

[5] Winkels, R., J. Sandberg, and J. Breuker. 1990. The Coach. Iz *EUROHELP: Developing Intelligent Help Systems*, Breuker, J., ed., EC: Copenhagen

[6] Pepite web site:

[7] Minerva web site: