A Framework for Developing Intelligent Tutoring Systems Incorporating Reusability

Eman El-Sheikh and Jon Sticklen

Appears in:

El-Sheikh, E., and Sticklen, J. (1998). A Framework for Developing Intelligent Tutoring Systems Incorporating Reusability. IEA-98-AIE: 11th International Conference on Industrial and Engineering Applications of Artificial Intelligence and Expert Systems, Benicassim, Castellon, Spain, Springer-Verlag (Lecture Notes in Artificial Intelligence, vol. 1415).

A Framework for Developing Intelligent Tutoring Systems Incorporating Reusability

Eman El-Sheikh and Jon Sticklen

Intelligent Systems Lab, Computer Science Department, Michigan State University, 3115 Engineering Building, East Lansing, MI 48824 USA
{elsheikh, sticklen}@cps.msu.edu

Abstract. The need for effective tutoring and training is mounting, especially in industry and engineering fields, which demand the learning of complex tasks and knowledge. Intelligent tutoring systems are being employed for this purpose, thus creating a need for cost-effective means of developing tutoring systems. We discuss a novel approach to developing an Intelligent Tutoring System shell that can generate tutoring systems for a wide range of domains. Our focus is to develop an ITS shell framework for the class of Generic Task expert systems. We describe the development of an ITS for an existing expert system, which serves as an evaluation test-bed for our approach.

1 Introduction

The need for effective tutoring and training is rising, given the increasing complexity of the work place, and the knowledge-drain in contemporary commercial settings. This is especially true in industry and engineering fields. Rapid progress in science and technology has created a need for people who can solve complex problems and operate and maintain sophisticated equipment. Many Computer-Assisted Instruction (CAI) techniques exist that can present instruction, and interact with students in a tutor-like fashion, individually, or in small groups. The introduction of Artificial Intelligence techniques and Expert Systems technology to CAI systems gave rise to Intelligent Tutoring Systems (ITSs), i.e., intelligent tutors that can model the learner's understanding of a topic and adapt the instruction accordingly. Although ITS research has been carried out for over 15 years, few tutoring systems have made the transition to the commercial market. The main reasons for this failure to deliver are that the development of ITSs is difficult, time-consuming, and costly. There is a need for easier, more cost-effective means of developing tutoring systems.

In this paper, we describe a novel approach to developing an Intelligent Tutoring System shell that can generate tutoring systems for a wide range of domains. Our focus is to develop a family of ITS shells for the class of Generic Task (GT) expert systems. The goal is to develop an ITS architecture that can interact with any GT-based system, and produce an effective tutorial covering the domain knowledge

represented in that system. As a test-bed for our approach, we describe the development of an ITS for the composite material design domain from a composite material design system that was built using the GT analysis and implementation methodology. An evaluation of the test-bed development project reveals the strengths and weaknesses of our approach.

The remainder of this paper describes the current state of our work on this topic, and constitutes a progress report. The next section highlights the problems with the traditional ITS approach and the motivation behind our approach. Section 3 details the background theories we build upon and describes related research and development efforts. In section 4, we present our approach for developing reusable ITSs. Finally, section 5 offers an evaluation of our results and a discussion of the contributions and future work.

2 Problem and Motivation

2.1 The Problem

A serious problem exists in the current methodology of developing Intelligent Tutoring Systems. Each application is developed independently, and tutoring expertise is hard-coded into individual applications. There is very little reuse of tutoring components, such as the student model, tutoring model, and user interface, between applications because we lack a standard language for representing the knowledge, a standard interface to allow applications to access the knowledge, and a set of tools to allow designers to manipulate the knowledge. In describing the state of building ITSs, Clancey and Joerger [4] lament that "...the reality today is that the endeavor is one that only experienced programmers (or experts trained to be programmers) can accomplish. Indeed, research of the past decade has only further increased our standards of knowledge representation desirable for teaching, while the tools for constructing such programs lag far behind or are not generally available."

2.2 Motivation for our Work

The motivation for our work comes from the need for reusable Intelligent Tutoring Systems and from the leverage that the Generic Task development methodology offers in solving this problem. The assumption of the GT approach is that there are basic "tasks" - problem solving strategies and corresponding knowledge representation templates - from which complex problem solving may be decomposed. Our goal is to develop an ITS architecture that can interact with any GT-based system, and produce an effective tutorial covering the domain knowledge represented in the problem solver. The backing intuition of this work is that GT systems are strongly committed to both a semantically meaningful knowledge representation

method, and to a structured inferencing strategy, and by leveraging this strong structure, automated generation of tutorial overlays are enabled.

This approach facilitates the reuse of tutoring components for various domains. The ITS shell can be used in conjunction with any GT-based expert system, effectively allowing the same tutoring components to be plugged in with different domain knowledge bases [6]. As a test-bed for our approach, we are working on the development of an ITS in the industrial and engineering domain of composite material design and fabrication. Our goal is to develop an ITS for this domain utilizing only the domain knowledge represented within a GT-based composite material design system.

3 Background and Related Work

3.1 Expert System Development Methodology: Generic Tasks

The idea of Generic Tasks can be understood at one level as a semantically motivated approach to developing reusable software - in particular reusable shells for knowledge-based system analysis and implementation. Each GT is defined by a unique combination of: (1) a well-defined description of GT input and output form, (2) a description of the knowledge structure which must be followed for the GT, and (3) a description of the inference strategy utilized by the GT [3]. To develop a system following this approach, a knowledge engineer first performs a task decomposition of the problem, which proceeds until a sub-task matches an individual generic task, or another method (e.g., a numerical simulator) is identified to perform the sub-task. The knowledge engineer then implements the identified instances of atomic GT building blocks using off-the-shelf GT shells by obtaining the appropriate domain knowledge to fill in the identified GT knowledge structure. Having a pre-enumerated set of generic tasks and corresponding knowledge engineering shells from which to choose guides the knowledge engineer during the analysis phase of system development.

A number of these atomic Generic Tasks are currently available, and are implemented in our toolset. These include Structured Matching, Hierarchical Classification [8] and Routine Design [2]. The Knowledge Level Architecture (KLA) [16] provides an organizational overlay to the basic Generic Task approach to facilitate integration. Routine Design (RD) is the GT type we are interested in here, being the structure used to develop the test-bed expert system. RD was proposed as an architecture for performing design and planning tasks in which substantial experience is available (not for design or planning in totally novel situations). The generic task approach is only one approach for expert system development. For an overview on other task-specific approaches, see [18] or [15].

3.2 Test-Bed System

As a test-bed for our approach, we are working with a GT system called COMADE (COmposite MAterial DEsigner) [11], which is utilized in the design phase of composite material applications. The inputs to the system are characteristics of the target application. The output is a set of designs satisfying these requirements. Composites are more flexible than corresponding metals because there is a very wide range of choices for the various components of a composite. COMADE designs a composite material by finding satisfying choices for each of these components.

3.3 A Theory of Learning

To develop an effective tutoring system, we need to focus on the needs of the student, and how to address those needs using instructional techniques. This calls for an understanding of how the student uses and learns from the domain knowledge offered by the tutor. At the psychological level of analysis of learning, emphasis is placed on the construction of personal knowledge and on a learner-centered approach to pedagogy [13]. This approach to learning is commonly referred to as constructivism.

There is a distinction among the types of knowledge to be learned. Declarative knowledge involves knowledge of facts, concepts, and vocabulary. Procedural knowledge is demonstrated when a student can combine declarative knowledge so that it can be used in a course of action. Strategic knowledge involves knowing when and how to use declarative and procedural knowledge to construct a learning outcome. This type of knowledge epitomizes the active construction of knowledge and places the student at the center of the teaching-learning process.

Learning from strategic knowledge construction suggests that instruction for skill acquisition should, for the most part, be given in a problem-solving context [17]. This way, declarative and procedural knowledge can be converted into useful learning outcomes immediately. Learning is enhanced because the problem-solving context provides a set of conditions encoding the applicability of the knowledge and its relevance to problem-solving goals. This is the idea of learning by doing.

3.4 Related Work

This section describes the commonly-adopted standard architecture of an ITS and related work on the development of tutoring systems shells. There is widespread agreement within the ITS community that an ITS consists of four “expert” modules, as depicted in figure 1 [14]. The expert module contains the domain knowledge. The student model diagnoses what the student knows. The tutor identifies which deficiencies in knowledge to focus on and selects the appropriate instructional strategies to present that knowledge. The user interface is responsible for communication with the learner. This ITS framework will be extended in section 4.

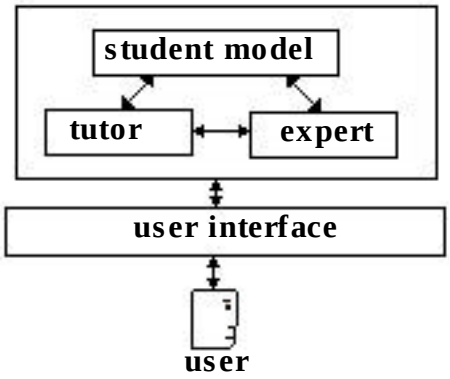

Fig. 1. Standard ITS architecture

Several other efforts are being made to develop ITS shells and reusable ITS components, although they are not discussed here due to space limitations. For more on these efforts, see [1, 7, 10, 12].

4 A Framework for Building Intelligent Tutoring Systems

4.1 Extending the Generic Task Framework

The goal of our work is to build an ITS framework that can interact with any GT-type problem solver to produce a tutoring system for the domain addressed by the problem solver. The learner interacts with both the tutoring system (to receive instruction, feedback, and guidance), and the expert system (to solve problems and look at examples), as shown in figure 2. Rather than re-implement the expert module for each domain, the ITS shell interfaces with a GT system to extract the necessary domain knowledge. This facilitates the reuse of the tutor, user interface, and student model components for different domains. Linking the ITS's expert module to the problem solver deserves special consideration. Rather than encode domain knowledge explicitly, the expert module extracts and tries to make sense of the domain knowledge available in the expert system. Thus, the quality of the tutoring knowledge is affected by the knowledge representation used by the expert system. The GT methodology's strong commitment to both a semantically meaningful knowledge representation method, and a structured inferencing strategy, allows the extraction of well-defined tutoring knowledge. The expert module can extract three types of knowledge: (a) decision-making knowledge (how the data relates to the knowledge), (b) knowledge of the elements in the domain knowledge base, and (c) knowledge of the problem solving strategy and control behavior.

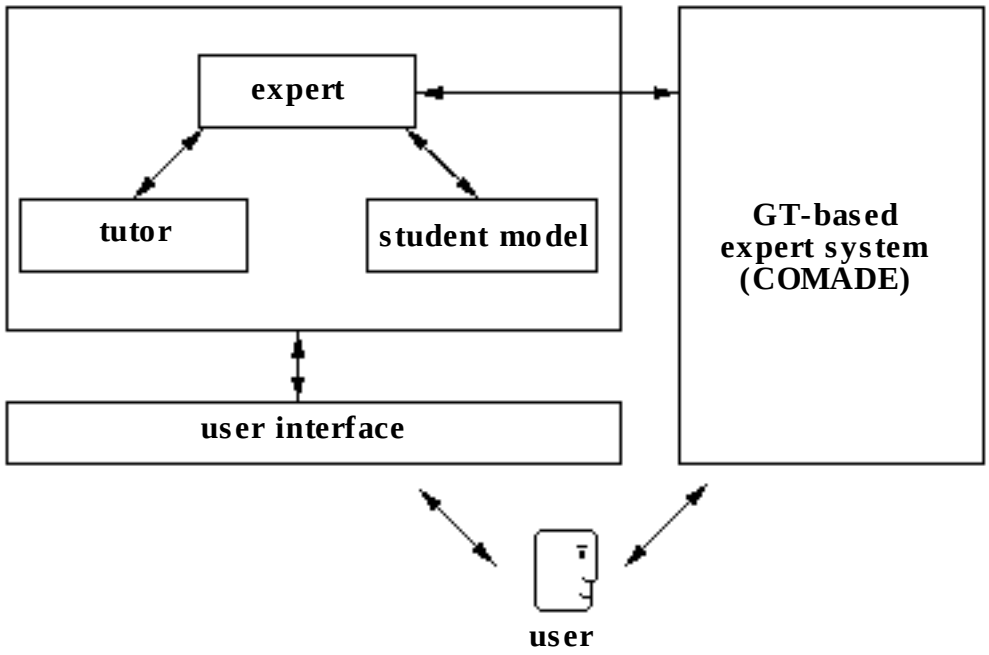

Fig. 2. System-user interaction model

To make the knowledge available to the ITS, the expert system must understand its own knowledge representation. The specific expert system used in this project, COMADE, is a Routine Design system, as described in section 3.1. We extend the structure of RD to allow any Routine Design system to have a self-understanding of its knowledge structures and reasoning processes. An RD system is composed of agents, each of which has a specific goal, purpose, and plan of action. For the extraction of tutoring knowledge, each agent must be able to answer three basic questions about itself:

1. WHAT did you do?
   The answer to this type of question is the result of the agent's action.
2. WHY did you do this?
   This is answered using a description of the agent's context (from its super-agent).
3. HOW did you do it?
   The answer is a description of the agent's actions, possibly including those of its sub-agent(s).

The expert module of the tutoring shell uses these answers, along with an encoding of a Routine Designer's structure to formulate domain knowledge as required by the ITS.

4.2 Completing the ITS Shell Framework

Thus far, we have explained the key component of the ITS shell that allows reuse, namely the expert module, and how it interfaces to a GT-based expert system. To complete the shell, this module must work in conjunction with a tutoring module and student model to interact with the learner. The main focus of this paper is to describe how reusability is achieved, but here we will briefly describe the other components.

Student modeling is the key to individualized knowledge-based instruction [9]. Within the context of the ITS, the student modeler plays an important role, in that it allows the tutoring system to adapt to the needs of individual learners. The approach we adopt for student modeling that seems to have potential is through the use of function-based reasoning. A functional model is used to simulate the learner and provide a description of the student state. The student's goals are modeled as a hierarchical model of functions. The top-level function or goal of the student is to learn the domain knowledge embedded within the problem solver. This main goal (function) is decomposed into lower level goals (functions) within the hierarchical model, which may be further decomposed into sub-goals (sub-functions) and so on. Figure 3 shows the Information Processing Task (IPT) Model for the student modeling component. The student model uses information provided from the user, expert module, and instructional module, to keep an accurate model of the learner's knowledge level and capabilities, and also to guide the instructional strategy. Our approach has several benefits: it provides the potential for generating explanations of learner behavior, knowledge, and misconceptions, as well as explanations of the reasoning process. In addition, a functional model can provide causal reasoning capabilities, and thus map onto human reasoning techniques.

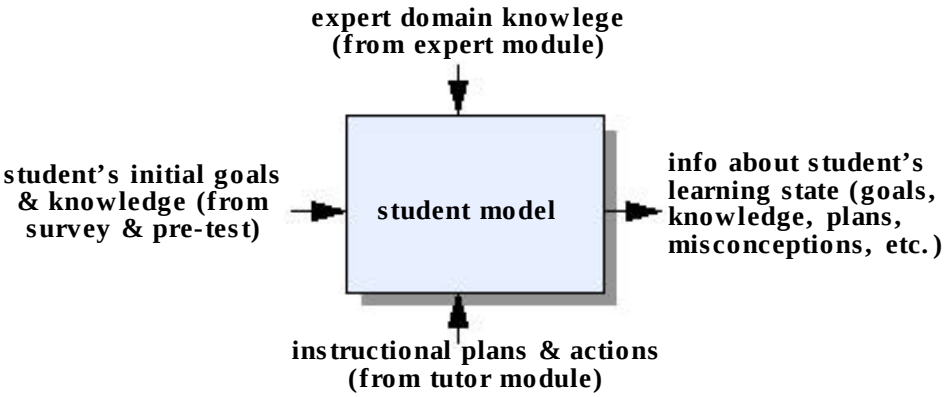

Fig. 3. IPT model for the student modeler

The tutor or instructional module uses two main instructional strategies, learning by doing and case-based teaching, in addition to question-answer templates. These teaching strategies are well-suited for teaching complex, knowledge-intensive domains, such as engineering domains. Moreover, they are a good match for our

framework, since the learner can interact with both the expert system to solve problems and the tutoring system. Learning by doing is implemented within the shell by having the learner solve real problems using the expert system, with the tutor watching over as a guide. In the other learning mode, the tutor makes use of the case-based knowledge base of the expert system, in which the input-output sets are stored as individual cases. The instructional module can present prototypical cases to the user, which serve as a basis for learning from new situations. Alternatively, it can present a new situation, posed as a problem, and ask for a solution. The goal is to help the user develop a set of cases, and determine their appropriateness within the domain.

4.3 The COMADE Example

In this section, we present a description of how the user interacts with both the tutoring system and the expert system, a composite material designer called COMADE. There are two modes of interaction with the system. The first is tutor-driven and the instruction combines learning by doing and case-based teaching. In this mode, the tutor can either present and explain a case to the user, or ask the user to solve a case. This type of instruction is performed at two different levels of tutoring knowledge. At a low level, a case covers individual knowledge constructs (e.g., facts, concepts, processes). This allows the tutoring of declarative and procedural knowledge. At a higher level, a case consists of a complete knowledge set to solve the top-level problem addressed by the expert system. In the case of COMADE, a high-level case would include all the knowledge and reasoning required to design a composite material satisfying certain input requirements. This type of case facilitates strategic knowledge learning by the user. Example interactions are shown in figures 4 and 5.

```
Tutor: To develop a composite material application with
properties: required flexural modulus = 40 GPa,
required tensile strength = 2000 MPa, flame retardance
= no,... {other requirements}, we can use the following
material options: use Polyamide-66 as the matrix type,
P100 graphite as the fiber type, a continuous fiber
length, and no chemical agent. This choice was made
because...{explanation of the reasoning process for the
solution path}.
```

Fig. 4. Example of tutor-driven interaction: presenting a high-level case

```
Tutor: Given the following design requirements:
required flexural modulus is 420 GPa, and required
tensile strength is 1390 MPa, what type of fiber is
most appropriate?

User: P-100 Graphite seems the most appropriate.
```

```
Tutor: Correct.
```

Fig. 5. Example of tutor-driven interaction: solving a low-level case

The second mode of system interaction is user-driven, in the form of question-answer sessions. In this mode, the user chooses what he/she wants to learn about, by asking the system questions. The tutor can answer three types of questions: (1) decision-making questions, (2) questions about the elements in the domain knowledge base, and (3) questions about the problem solving strategy and control behavior. Figure 6 shows examples of question-answer sessions for the three types of questions.

```
User: Why did you choose a glass material as the fiber
reinforcement? (Type 1)

Tutor: I chose E-glass because the required flexural
modulus is 40 GPa, the required tensile strength is
2000 MPa, and the chemical environment is not an acid.
E-glass is an appropriate fiber for these requirements.

User: What is the appropriate range of use temperature
for a Polyamide-66 material? (Type 2)

Tutor: The appropriate use temperature for Polyamide-66
is between 120 - 140 C.

User: What does the reinforcement plan do? (Type 3)

Tutor: In the context of determining an appropriate
reinforcement for the composite material, the
reinforcement plan selects an appropriate fiber
material and length.
```

Fig. 6. Example question-answer sessions

5 Results and Discussion

We have presented the framework of a shell for building Intelligent Tutoring Systems for various domains by interfacing with existing expert systems, and reusing the other tutoring components. Specifically, we have described the development of an ITS for the composite materials design domain. More generally, we have formulated a technique for leveraging the knowledge representation and structure of the Routine Design framework for tutoring. And we have laid the foundation of our top-level goal - to develop a framework for an ITS extension to the GT approach. The main problem we addressed is reusability. We have found a solution to this

problem in the development of tutoring systems for existing expert systems and knowledge bases.

Further research is needed to make the framework more concrete. Future research paths include developing an ontology of the tutoring knowledge available within a GT-based expert system. We also hope to test our approach further by applying it to other existing expert systems. Testing our ITS shell using several expert systems will provide more insight into the benefits and limitations of our approach.

References

1. Antao, B., Broderson, A., Bourne, J., and Cantwell, J.: Building intelligent tutorial systems for teaching simulation in engineering education. *IEEE Transactions on Education*, 35(1), pp. 50-56 (1992)
2. Brown, D.: Design. *Encyclopedia of AI*, 2nd edition (S. Shapiro, ed.), Wiley-Interscience Publishers, New York, NY (1991)
3. Chandrasekaran, B.: Generic tasks in knowledge-based reasoning: high-level building blocks for expert system design. *IEEE Expert*, 1(3), pp. 23-30 (1986)
4. Clancey, W., and Joerger, K.: A practical authoring shell for apprenticeship learning. *Proc. of ITS'88: First Intl. Conf. on Intelligent Tutoring Systems*, Montreal, Canada, pp. 67-74 (1988)
5. Costa, E.: *New Directions for Intelligent Tutoring Systems*, Springer-Verlag Publishers, New York, New York (1991)
6. El-Sheikh, E., Kamel, A., and Sticklen, J.: An ITS shell leveraging the generic task approach to problem solving. *Proc. of the ITS'96 Workshop on Architectures and Methods for Designing Cost-Effective and Reusable ITSs*, Montreal, Canada (1996)
7. Fleming, J., and Horwitz, C.: Applications of the rapid intelligent tutoring system development shell. *Proc. of the ITS'96 Workshop on Architectures and Methods for Designing Cost-Effective and Reusable ITSs*, Montreal, Canada (1996)
8. Gomez, F., and Chandrasekaran, B.: Knowledge organization and distribution for medical diagnosis. *IEEE Transactions on Systems, Man, and Cybernetics*, SMC-11(1), pp. 34-42 (1981)
9. Holt, P., Dubs, S., Jones, M., and Greer, J.: The state of student modelling. In: *Student Modelling: The Key to Individualized Knowledge-Based Instruction* (J. Greer and G. McCalla, eds.), Springer-Verlag Publishers, Berlin, Germany, pp. 3-35 (1994)
10. Kameas, A.: The functional architecture and interaction model of a generator of intelligent tutoring applications. *Journal of Systems Software*, 36, pp. 233-245 (1997)
11. Lenz, T., McDowell, J., Moy, B., Sticklen, J., and Hawley M.: Intelligent decision support for polymer composite material design in an integrated design environment. *Proc. of the American Society of Composites 9th Technical Conference*, pp. 685-691 (1994)
12. Murray, T.: Having it all, maybe: design tradeoffs in ITS authoring tools. *Proc. of ITS'96: Third Intl. Conf. on Intelligent Tutoring Systems*, Montreal, Canada, pp. 93-101 (1996)
13. Phye, G.: Learning and remembering: the basis for personal knowledge construction. In: *Handbook of Academic Learning: Construction of Knowledge* (Gary Phye, ed.), Academic Press, San Diego, California, pp. 47-64 (1997)

14. Polson, M. C., and Richardson, J. J.: *Foundations of Intelligent Tutoring Systems*, Lawrence Erlbaum Publishers, Hillsdale, New Jersey (1988)
15. Steels, L.: Components of expertise, *AI Magazine*, 11(2), pp. 28-49 (1990)
16. Sticklen, J.: Problem solving architecture at the knowledge level. *Journal of Theoretical and Applied Artificial Intelligence*, 1, pp. 1-52 (1989)
17. Wenger, E.: *Artificial Intelligence and Tutoring Systems: Computational and Cognitive Approaches to the Communication of Knowledge*, Morgan Kaufmann Publishers, Los Altos, California (1987)
18. Wielinga, B.: Schreiber, A., and Breuker, J., KADS: A modeling approach to knowledge engineering. *Knowledge Acquisition*, 4(1), pp. 5-54 (1992)