Position paper for

AAAI Fall-97 Symposium on ITS Authoring Tools:

Knowledge Based Tutor Authoring Tools

Tom Murray

Center for Knowledge Communication
Dept. of Computer Science
University of Massachusetts Amherst, MA 01003
tmurray@cs.umass.edu, www.cs.umass.edu/~tmurray

RESEARCH BACKGROUND

My focus has been on designing systems that allow practicing educators and instructional theorists to participate more completely in the design of advanced technology instructional systems (including intelligent tutoring systems and knowledge based tutors). My research interests include the representation and acquisition of instructional expertise and subject matter knowledge, interoperability and reusability in computer based instruction, and shared vocabularies and ontologies. Currently I am managing an R&D project at the University of Massachusetts Center For Knowledge Communication. The goal of the project is to produce an authoring system for knowledge-based multimedia instruction. The system, called Eon, facilitates the creation of generic instructional strategies and the re-use of instructional content in building highly interactive teaching systems which adapt to the pedagogical needs of the student. (see http://www.cs.umass.edu/~tmurray/eon_www/eon.html for the Eon project home page, and http://www.cs.umass.edu/~tmurray/papers.htm for a list of downloadable publications). Recently I have also been working on models for "distributed curricula" which enable tutorial resources to intelligently invoke each other over the WWW.

THE EON ITS AUTHORING TOOLS

Intelligent Tutoring Systems (ITSs) are computer-based instructional systems that have separate data bases, or knowledge bases, for instructional content (specifying what to teach), and for teaching strategies (specifying how to teach), and attempt to use inferences about a student's mastery of topics to dynamically adapt instruction. ITSs, also called knowledge based tutors, use techniques that allow automated instruction to come closer to the ideal than traditional computer based instruction by more closely simulating realistic situations, and by incorporating computational models (knowledge bases) of the content, the teaching process, and the student's learning state.

In the last half decade ITSs have moved out of the lab and into classrooms and work places where some have proven to be highly effective as learning aides [Shute and Region 1990]. While intelligent tutoring systems are becoming more common and proving to be increasingly effective, each one must still be built from scratch at a significant cost. Little is available in terms of authoring tools for these systems. Authoring systems are commercially available for traditional CAI and multimedia-based training, but these authoring systems lack the sophistication required to build intelligent tutors. Commercial authoring systems excel in giving the instructional designer tools to produce visually appealing and interactive screens, but behind the presentation screens is a shallow representation of content and pedagogy.

Our approach to developing ITS authoring tools is motivated by issues of pragmatics and usability. Rather than starting with a laboratory-based AI tutoring system and asking how it can be generalized to produce more generic shell, as is the case in the design of some ITS authoring shells, our design base-line is commercially available and widely used authoring systems for traditional computer-based teaching, such as Authorware and Icon Author. Our goal was to extend rather than replace the capabilities afforded by such systems, to preserve the level of usability and some of the authoring methods instructional designers have become familiar with, and add additional tools, features and authoring paradigms to allow more powerful and flexible tutors to be built.

There is a large established user population using COTS (common off the shelf software) to create instructional software. We would like to facilitate cost-effective ITS production by teachers and instructional designers, as well as by those who have traditionally built ITSs from scratch (primarily those in academic and industrial research labs). Empowering instructional designers to build these more powerful systems requires new tools and a paradigm shift in

the way many of them conceptualize instructional systems. However, we want this shift to be accessible, incremental, and evolutionary. For this reason, on the surface many of our tools have a look and feel similar to COTS tools, yet allow additional levels of abstraction, modularity, and visualization necessary for producing an ITS.

Specifically, we proposed that moving from CAI authoring to ITS authoring involves a fundamental paradigm shift from "story board" representations of instructional material to more powerful and flexible "knowledge based" representations. The basic concept is not new; in fact, it is fundamental to all AI work. Our contribution is in fleshing out how the knowledge based paradigm can be best presented to empower instructional designers.

In traditional CAI instructional actions are encoded using building blocks at the level of the media: text, pictures, button clicks, etc. In contrast, knowledge based tutors can facilitate the design of instructional actions using pedagogically relevant building blocks. For example: "give a hint," or "teach the prerequisites". Designing instruction using building blocks such as "hint," "prerequisite," "if-confused," "known," and "summarize" is much more powerful than designing instruction at the level of "show video," "present picture" or "wait for the button click" The instructor can conceptualize the curriculum at a more natural level of abstraction. An instructional strategy in the intelligent tutor might be: "if the current topic is conceptual and the student is doing poorly, give several examples." Alternate strategies can be created, so that the appropriate strategy can be used according to the needs of the student (e.g. learning style or mastery of the current topic) or the pedagogical characteristics of the content being taught (e.g. whether it is procedural or conceptual information).

A CURRICULUM OBJECT FRAMEWORK

The Eon system uses the following objects and mechanisms to in its representational formalism:

Topics. Knowledge elements of any grain size are represented by Topics. Since one can define specific types of topics (see below) and create hierarchical links between topics, this simple object suffices for many representational schemes. Topics can have any number of Topic Properties, such as difficulty and importance.

Topic Links. Customizable link types allows for the representation of a wide variety of topic networks, including component hierarchies, skill lattices, concept networks, etc.

Presentation Contents. Whereas Topics are abstract objects that refer to modular chunks of the knowledge to be taught, Presentation Contents (or just "Contents") contain the specific contents that the student will see and manipulate (text, graphics, etc., or the templates or algorithms for generating this content). Contents are expository or inquisitory interactions, usually associated with a specific interactive screen templates which the author designs using the Interaction Editor.

Topic Levels. Having only the semantic net to represent all aspects of curriculum structure was found inadequate. Topic Levels allow for distinguishing multiple levels of performance (e.g. memorizing vs. using knowledge), mastery (novice to expert), and pedagogical purpose (summary, motivation, example, evaluation, etc.) for each topic.

Topic Types. As mentioned, there is general agreement that there are different types of knowledge, each type having its own properties and each type requiring a different instructional method. Rather than have knowledge type be simply a property of Topics, Topic Types are first class objects which all Topics inherit from. Since there are numerous theories of knowledge types, we leave knowledge type definition to the Ontology.

Lessons. Topic networks, which define pedagogically relevant relationships between topics, do not specify any ordering or starting point for learning sessions, and are independent of the instructional purpose of sessions. Lesson objects are used to specify instructional goals and learning/tutoring styles for a particular group of students.

The basic representational elements described above were included in Eon because some form of all of them (with the exception of Topic Levels) seem necessary for any pedagogical representation. We call this framework a "five layer decision architecture," where the five layers are Lesson, Topic, Topic Level, Content, and Events. Running a Lesson runs a number of Topics, each of which runs some of its Topic Levels, each of which contains a number of Contents. At the lowest level are the individual Events between the student and tutor, such as a student clicking a button, or the tutor giving a hint, several of which will occur while a Content is running.

The five-layer decision architecture is fixed in Eon, but the specifics of each layer are customizable using ontology objects. An ontology is a particular way of describing the world (or some domain); it is a scheme for conceptualizing the objects and relationships in a domain. We use the term "ontology object" for a data object which defines a conceptual vocabulary for a part of the system. In our current system Topic Ontology objects specify topic types, topic link types, topic properties, and topic levels. For example, the Topic Ontology for our Statics Tutor defines topic types Fact, Concept, Procedure, and Misconception, and topic links Prerequisite, Generalization, and SubConcept; while a tutor for Manufacturing Equipment might have topic types Safety, Maintenance, Operation, Theory, and Common Failures, and topic links SubPart and SimilarPart. Ontologies tend to be generic and reusable, for

example, an ontology developed for one science tutor should be usable (perhaps with slight modifications) for other tutors with similar pedagogical characteristics (e.g. instruction at a predominantly conceptual level).

AUTHORING TOOLS

We have built a suite of prototype authoring tools that support the knowledge based paradigm for designing instructional systems. The goal the Eon tools is to provide a suite of authoring tools that are as intuitive and usable as commercially available tools, yet support designers in making the leap from the story board paradigm to the knowledge based paradigm. We will briefly describe several of the tools in Eon below.

The Interaction Editor allows trainers to construct student screens, student interactions, simulations, learning environments, and multimedia materials. The "widget palette" shown to the far left allows for the creation of over a dozen widgets, including sliders, movies, hot spots, and tables. What distinguishes the screens in Eon from those in CAI authoring tools is that the screens are templates. For instance, the screen shown in the picture is a generic template for multiple choice questions with a picture, a picture description, and a confidence slider. The author creates a number of "Presentation Contents" to fill in this template with specific values, i.e. specific text and pictures for each use of the generic screen.

As the author builds the interactive screens, a data base-like template is automatically created showing the relevant parameters of each widget. The Presentation Contents Editor shows the parameters for each widget on the screen, and is used to create and edit Presentation Contents. The Figure shows the contents for "FlyOnATable," one of the many Presentations created for this template. As can be seen in the Figure, some widgets, such as graphics and text, have only one parameter (the name of the graphic, or the text string), while others have several (i.e. the multiple choice widget has the answers, the question text, and which answers are correct). The values of widget parameters can be bound to scripts or functions, allowing the student screens to be generated dynamically as well as contain canned material.

The Topic Network Editor is an interactive graphical tool that allows trainers to design curriculum networks and represent relationships between domain concepts. Topics have a Type (such as fact, concept or procedure), Links to other topics (as shown in the network), Topic Levels (not shown in the network, and described in [Murray 1996B]), and Topic Properties. The graphical properties shape, color, pattern, border color, and border thickness are used to depict various topic properties, such as importance, difficulty, and knowledge type. Links can be of various types, as indicated by the link color. Objects called Ontologies are used to customize the representational infrastructure of the domain, by specifying, for example, the types of topics, topic links, and topic properties used in a given tutor (see [Murray 1996C]).

The Topic Network allows curriculum design at an abstract, pedagogically relevant level, a capability not provided by CAI authoring tools. The Presentation Contents represent the concrete material the student will see on the screen. A tool called the Topic Browser (not shown) is used to link Presentation Contents to Topics, e.g. to specify which presentations are given to teach the topic Gravity.

So far we have described how the knowledge base is constructed from objects such as Topics and Presentations, but have not indicated how the author specifies how to sequence these and react to student behavior. The Strategy Editor utilizes a flow chart-like paradigm for graphically representing procedures, which we call Strategies. Icons representing interactive screens, branching and looping, variable manipulation, etc., are dragged from the Strategy Icon Pallet (bottom center of the Figure) and dropped onto the Strategy flow line. One thing that distinguishes these flowlines from similar ones used in CAI authoring tools is that Eon strategies are real procedures, with input parameters, local variables, and returned values, which can be called recursively, resulting in a much more powerful and expressive programming paradigm.

Eon contains several other tools, including a Student Model Editor, which allows instructors to specify how student behavior is used to infer student knowledge states and mastery levels of the Topics. Decisions in the Strategies can thus be predicated on the student's state, as opposed to simply the answer correctness, as in CAI.

These tools are currently being used to build five tutors. The Statics Tutor teaches introductory Statics concepts, and includes a "crane boom" simulation. The Bridging Analogies Tutor incorporates a Socratic teaching strategy developed and tested by cognitive scientists to remediate common persistent misconceptions in science. The Chemistry Workbench tutor provides a more open ended learning environment for learning about solvency and chemical reactions by interactively mixing chemicals and measuring the results. The Keigo Tutor teaches a part of Japanese language called "honorifics," dealing with the complicated rules used to determine verb conjugation which appropriately honors the listener and topic of a conversation. For this tutor Eon is interfaced with a rule based expert system. The Refrigerator Tutor explains the thermodynamic principles underlying refrigeration. All of these systems are in prototype stages.