Multiple Knowledge Sources in Intelligent Teaching Systems

Beverly Woolf
Patricia A. Cunningham

COINS Technical Report 87-72

This work was supported by the Air Force Systems Command, Rome Air Development Center, Griffiss Air Force Base, New York 13441-5700, and the Air Force Office of Scientific Research, Bolling AFB, DC 20332 under Contract No. F30602-85-C-0008. This contract supports the Northeast Artificial Intelligence Consortium (NAIC).

Multiple Knowledge Sources in Intelligent Teaching Systems

Beverly Woolf, University of Massachusetts
Patricia A. Cunningham, The Hartford Graduate Center

Intelligent teaching systems are emerging as a possible solution to the nation's large training problem in government, academy, and industry. But since the few systems that have been built were customized for specific applications, few guidelines or tools exist for building intelligent teaching systems. As a result, building such systems remains a black art—a pretechnology requiring considerable experimentation and effort while producing minimal results. In addition, existing systems show intelligence within a narrow specialization. Like other expert systems, tutoring systems display varying levels of expertise.[1] In every case, an intelligent tutor has less breadth of scope and flexibility than does its human counterpart. Incorporating community expertise could increase the limited competence of current tutors.

Model: *Mother Theresa, 1981.*

We believe that intelligent tutoring systems can be designed using a community memory, with multiple experts contributing their teaching and learning knowledge. The concept of knowledge base as community memory reflects the fact that knowledge is often distributed, incomplete, and acquired incrementally—especially in tutoring systems where the domain expert, cognitive scientist, and teaching expert are typically not the same person.[1] Experience with commercially successful expert systems such as R1[2] and the Dipmeter Advisor[3] suggests that using knowledge from a single expert can produce systems foreign to other system users—systems having conceptual holes. In the case of the Dipmeter Advisor, the expert model solved problems in an uncommon way, creating blind spots in the knowledge base.[1] To develop a community memory for tutoring systems, we need to create a framework within which

An earlier version of this article appears in the *Proceedings of the American Association of Artificial Intelligence*, Seattle, Wash., July 1987.

0885-9000/87/0500-0041 $01.00©1987 IEEE

teaching experience and domain wisdom can be saved. We must be able to (1) modify this framework, and (2) augment it as time passes.

We will study the development of three intelligent tutors, describing how lessons about acquiring knowledge from multiple experts were applied toward building each system. In particular, we will look at tools and methodologies for knowledge acquisition that might be transferred to other systems. We will first describe the role of computers in education, and then examine our three case studies in detail. Finally, we will extrapolate from these systems possible tools and criteria for building intelligent tutors.

Computers in education

Education is in trouble. Tough classroom problems, lack of public support, and inappropriate policies have left educators engaged in an uphill battle to reverse deficiencies in student learning and teacher training. This dilemma has fostered sometimes unrealistic expectations on the state of intelligent tutoring systems; hardware and software potential have led to exaggerated stories about the possible achievements of an as yet undeveloped technology.

As Figure 1 illustrates, innovation in computer technologies can be viewed as an S-curve measuring the relationship between the effort put into improving a product or process and the results one gets back from that investment.[4] Preintelligent tutoring systems, or computer-aided instructional systems, are at the end of their S-curve (Figure 1a); consequently, increased effort brings little additional performance. Expert systems (Figure 1b) are beginning to emerge from the low part of their S-curve as uniform and easy-to-use technologies like AI shells are developed, speeding production and performance. Intelligent tutoring systems (Figure 1c) are at the beginning of their S-curve; thus, they require substantial effort and new tools before they can produce substantial results.

Preintelligent tutoring systems achieved high performance in many areas by encoding built-in commitments regarding how their knowledge would be used.[5,6] These systems represented knowledge implicitly within the specific command that determined (and thus predefined) system input/output. As a result, each system required excessive development time.

The first hour of computer-aided instruction typically required about 200 hours of programmer preparation—an amount possibly exceeded by the preparation time required for building intelligent teaching systems. However, each additional hour of instruction for preintelligent systems also required about 200 hours of programmer preparation. Domain or teaching knowledge must be implicitly defined in the very question or explanation for which the knowledge is intended. This led to systems that could not automatically query students about concepts that had been previously defined.

In comparison, intelligent tutors are designed to make their knowledge explicit and uncommitted, thereby utilizing a single piece of knowledge in many ways. For example, by explicitly representing a concept such as force or acceleration, a physics tutor should have some flexibility in the use of that knowledge—and should be able to test students about the concept, describe the concept, demonstrate it within a simulation, and answer questions about it.

Case studies

The following three example tutors all required multiple sources of knowledge expressing their knowledge explicitly. From these case studies, we will attempt to derive tools and methodologies that can be used to acquire knowledge for the next intelligent tutor generation.

RBT for teaching complex industrial processes. The first tutor is fully implemented, tested, and now being used for training in nearly 60 industrial sites across America. It is still being augmented based on formal evaluations of student response to the system. RBT (the recovery boiler tutor) was built for a kraft recovery boiler—the type of boiler found in paper mills throughout the United States.[7]

Built by the J.H. Jansen Company and sponsored by the American Paper Institute, RBT provides multiple explanations and tutoring facilities tempered to the individual user (a control room operator). The tutor is based on a mathematically accurate formulation of the boiler and provides an interactive simulation complete with help, hints, explanations, and tutoring (see Figure 2).

Students can initiate any of 20 training situations, emergencies, and operating conditions—or they can ask that an emergency be chosen for them. Students can also accidentally trigger an emergency as a result of their actions on the boiler. Once an emergency has been initiated, students are encouraged to adjust

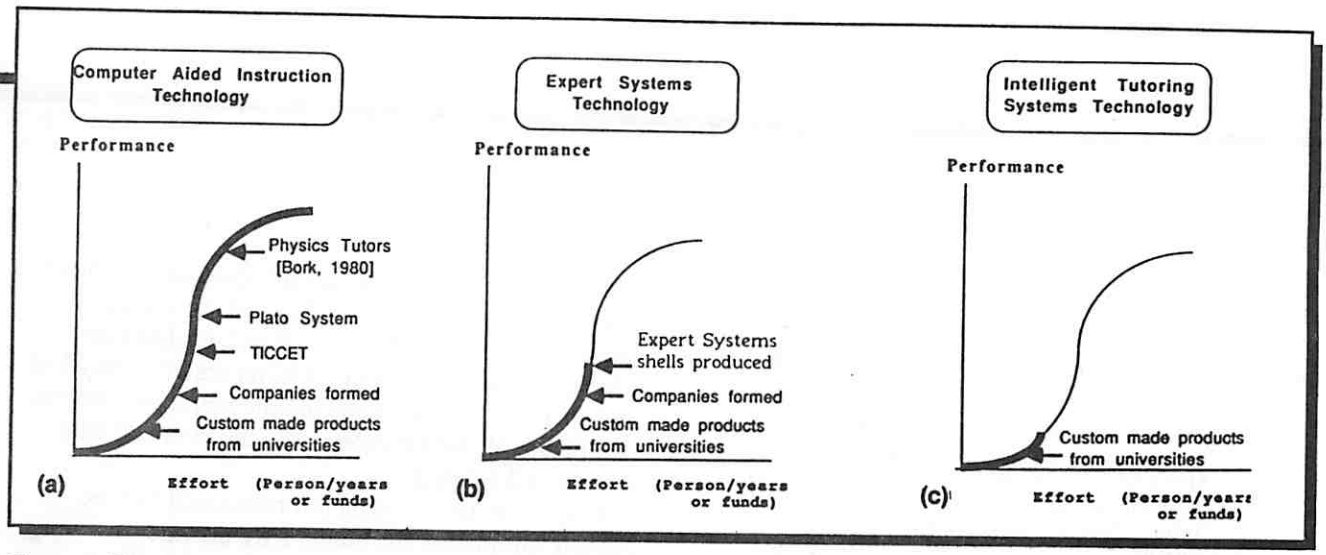

Figure 1. The S-curve of three computer technologies: **(a)** computer-aided instruction technology; **(b)** expert systems technology; **(c)** intelligent tutoring systems technology.

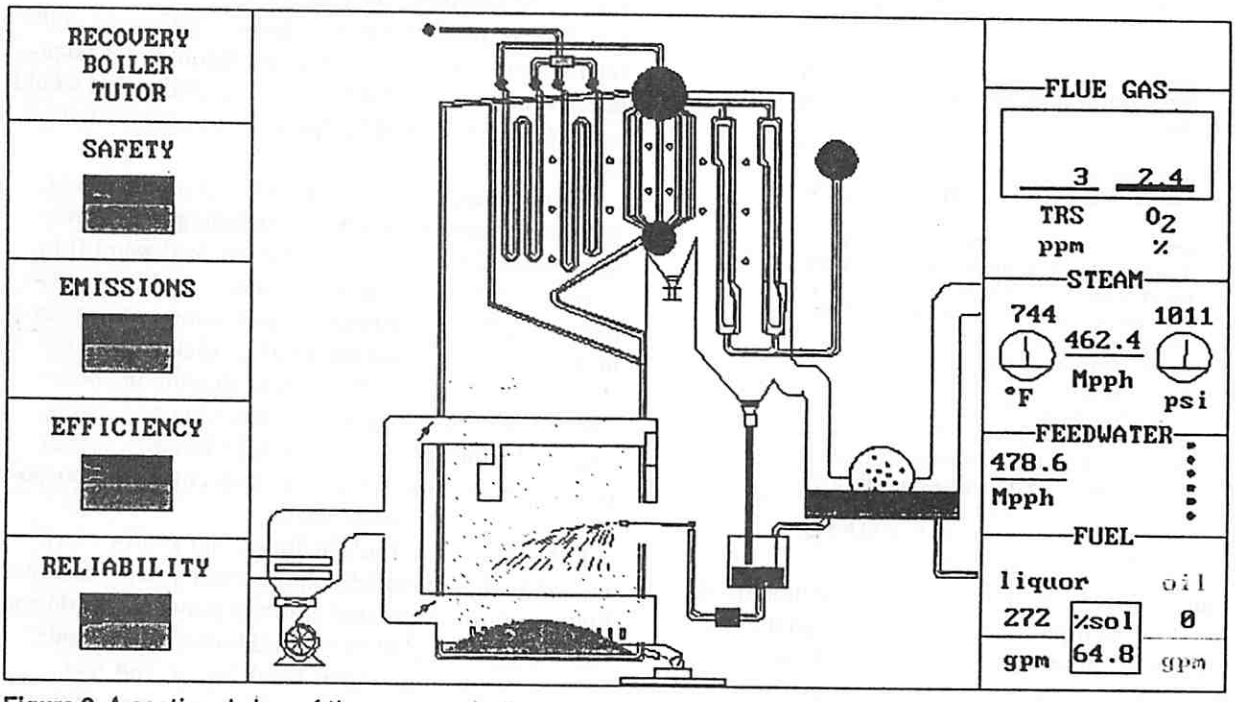

Figure 2. A sectional view of the recovery boiler.

meters and perform actions on the simulated boiler to solve the emergency. The system challenges student operators to solve new problems while the system monitors and advises those operators. RBT can recognize optimal, less than optimal, and clearly irrelevant actions. Operators can continue their freewheeling or purposeful problem-solving behavior while the tutor offers help, hints, and explanations—plus providing tutoring advice when needed or requested. Operators are expected to observe the impact of their actions on the simulated boiler and to react before the tutor advises them regarding potential problems.

An example dialogue. In the example interaction below between student and tutor, a partial blackout is in progress that will lead to combustion process deterioration if no action is taken. As students change set-point controllers and request information about the boiler, the tutor selectively discusses the optimality of their actions and suggests how they might better focus their actions or utilize their data. An important feature to note about this dialogue is that at any point during the simulated emergency there are many actions that operators might take and, as the problem worsens, an increasing number of actions that they *should* take to correct the operating conditions. Thus, an immediate and correct response might require only one action, such as rodding the primary air ports. But a delayed response causes the situation to worsen, requiring the addition of auxiliary fuel.

Meters indicate that CO, SO2, and O2 readings are abnormally high. An alarm buzzer sounds, beginning the following dialogue:

> **Tutor:** What action are you going to take?
> **Student:** Check smelt spout cooling water. [The student's action does not result in a change in the boiler.]
> **Tutor:** It looks as though the liquor isn't burning . . .
> **Student:** Help. What is going on? [The student requests help.]
> **Tutor:** A partial blackout is occurring and this causes TRS, CO, SO2, and O2 to increase. Steam flow and steam temperature are dropping. I think part of the bed is not burning.
> **Student:** Rod the primary air ports. [A small improvement in the boiler readings can be observed.]
> **Tutor:** Your action is correct; however, at this time rodding alone is not effective . . .
> **Student:** Raise the liquor temperature. [The student's action does not reverse the emergency and a new alarm buzzer sounds.]
> **Tutor:** Rodding the air port apparently was not enough to undo the blackout.
> **Student:** Increase the oil fuel supply. [The student's action reverses the emergency.]
> **Tutor:** That solved the problem alright. Good thinking.
>
> Analysis of the problem: You had a partial blackout caused by plugged primary air ports and a cold bed. Partial blackouts can be effectively treated through a combination of rodding the primary air ports and adding more heat. The problem can be avoided by keeping the air ports clean.

This dialogue was not actually produced in natural language; student input was handled through menus and tutor output produced by cutting text from emergency-specific text files loaded when the emergency was invoked. Operator interactions are handled through a hierarchy of menus enabling such activities as checking for a tube leak or rodding the smelt spout as well as selecting the alarm board or control panel board.

While the simulation of the recovery boiler is running, operators can view the boiler from many directions and can focus on several components; for example, the fire bed in Figure 3. The tutor provides assistance through visual clues such as a darkened smelt bed, acoustic clues, ringing alarm buzzers, textual help, explanations, and dialogues. Operators can request up to 30 process parameters on the complete panel board, view an alarm board (not shown), change 20 set points, and ask menu questions such as "What is the problem?" "How do I get out of it?" "What caused it?" and "What can I do to prevent it?" These questions are answered by cutting text from a file which was loaded with the specific emergency. These questions do not provide the basis of the tutor's knowledge representation, which is described elsewhere.[7] Operators can request meter readings, physical and chemical reports, and dynamic trends of variables (see Figure 4). All variables are updated in real time every 1 or 2 seconds.

In addition to providing information about the explicit variables in the boiler, RBT provides reasoning tools designed to aid students in reasoning about implicit processes in the boiler. One such tool is composite meters (shown on the left sides of Figures 2 through 5) recording the state of the boiler using synthetic measures for *safety, emissions, efficiency,* and *reliability* of the boiler. The meter readings are calculated from complex mathematical formulas that would rarely (if ever) be used by operators to evaluate the boiler.

For instance, the safety meter is a composition of seven independent parameters including steam pressure, steam flow, steam temperature, feed-water flow, drum-water level, firing-liquor solids, and combustibles in the flue gas. Meter readings allow students to make inferences about the effect of their actions on the boiler using characteristics of the running boiler. These meters are not presently available on existing pulp and paper mill control panels; however, if they prove effective as training aids, they could be incorporated into actual control panels.

Other reasoning tools include trend analyses (see Figure 5) and animated graphics such as shown in the Figures above. Animated graphics provide realistic and dynamic drawings of the several boiler components such as steam, fire, smoke, black liquor, and fuel. Trend analyses show how essential process variables interact in real time by allowing operators to select up to 10 variables including liquor flow, oil flow, and air flow—and to plot each against the others and against time.

Each student action, be it a set-point adjustment or a proposed solution, is recorded in an accumulated response value reflecting overall operator scores and how successful (or unsuccessful) operator actions have been and whether actions were performed in sequence with other relevant or irrelevant actions. This accumulated value is not presently used by the tutor, but the notation might be used to sensitize the tutor's future responses to student records. For instance, if operators have successfully solved a number of boiler emergencies, the accumulated value might be used to temper subsequent tutoring so that it is less intrusive. Similarly, if student performance has been poor, the

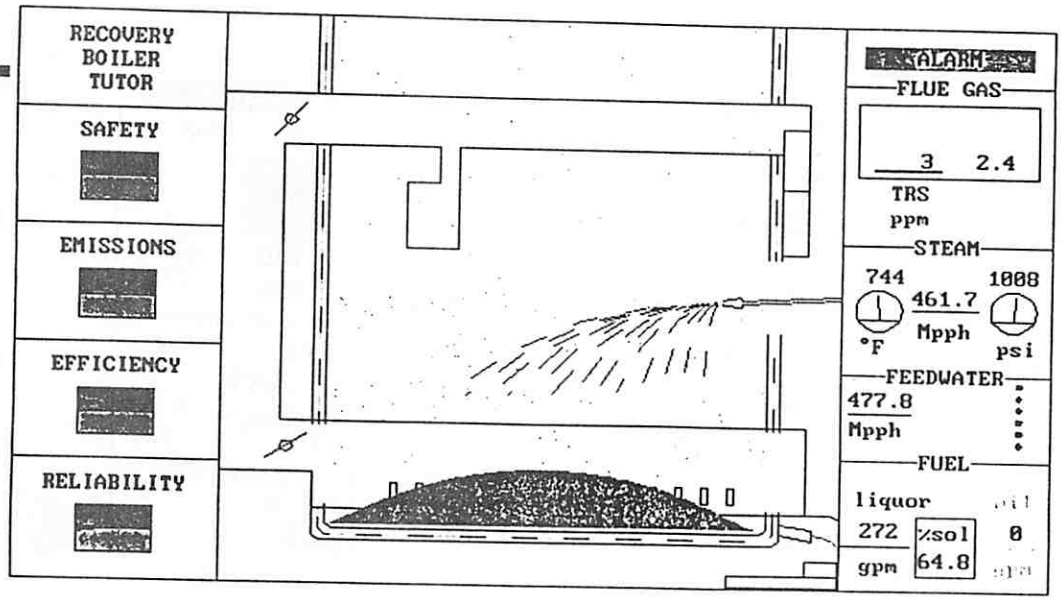

Figure 3. A focused view of the fire bed.

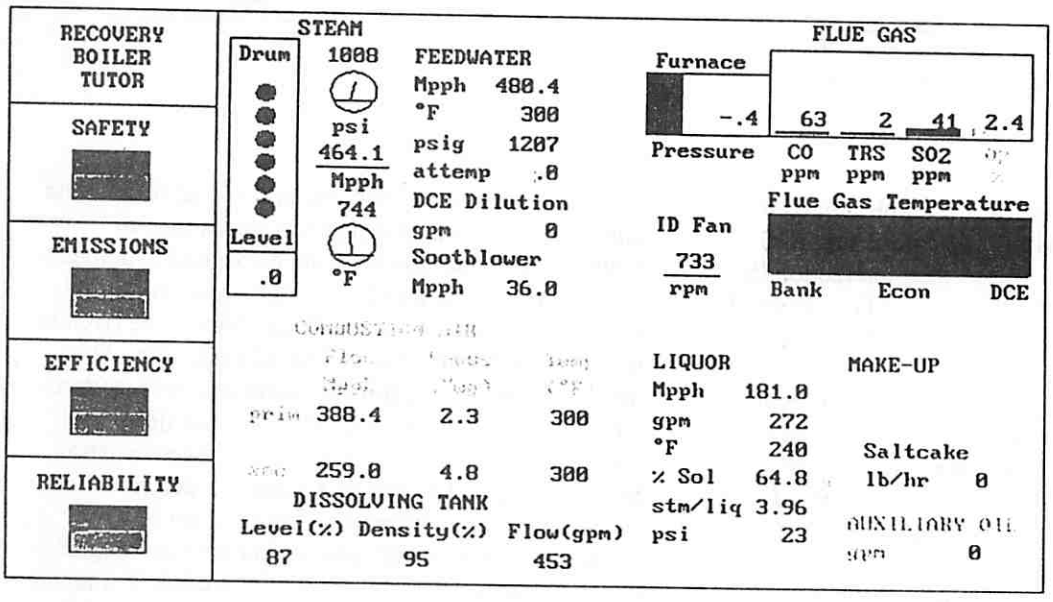

Figure 4. The complete control panel.

accumulated value could be used to activate more aggressive responses from the tutor.

RBT has been well received and is presently used in actual training in the control rooms of pulp and paper mills throughout the US. Formal evaluation has begun. However, informal evaluation suggests that operators enjoy the simulation and handle it with extreme care. They behave as they might in actual control of the pulp mill panel—slowly changing parameters, adjusting meters through small intervals, checking each action, and examining several meter readings before moving on to the next action. Experienced and novice operators alike engage in lively use of the system after about a half-hour introduction. When several operators interact with the tutor, they sometimes trade "war stories," advising each other about rarely seen situations. In this way, experienced operators frequently become partners with novice operators as they work together to simulate and solve unusual problems.

RBT was developed on an IBM PC AT with 512K-byte RAM, enhanced graphics, and a 20M-byte hard disk. It uses a math coprocessor, two display screens (one color), and a two-key mouse. The simulation was implemented in Fortran and took 321K bytes; the tutor was implemented in C and took 100K bytes. Although we tried to implement the tutor in Lisp, we found extensive interfacing and memory problems, including segment size restrictions (64k), incompatibility with the existing Fortran simulator, and addressable RAM restrictions (640K).

To circumvent these problems, the tutor was devel-

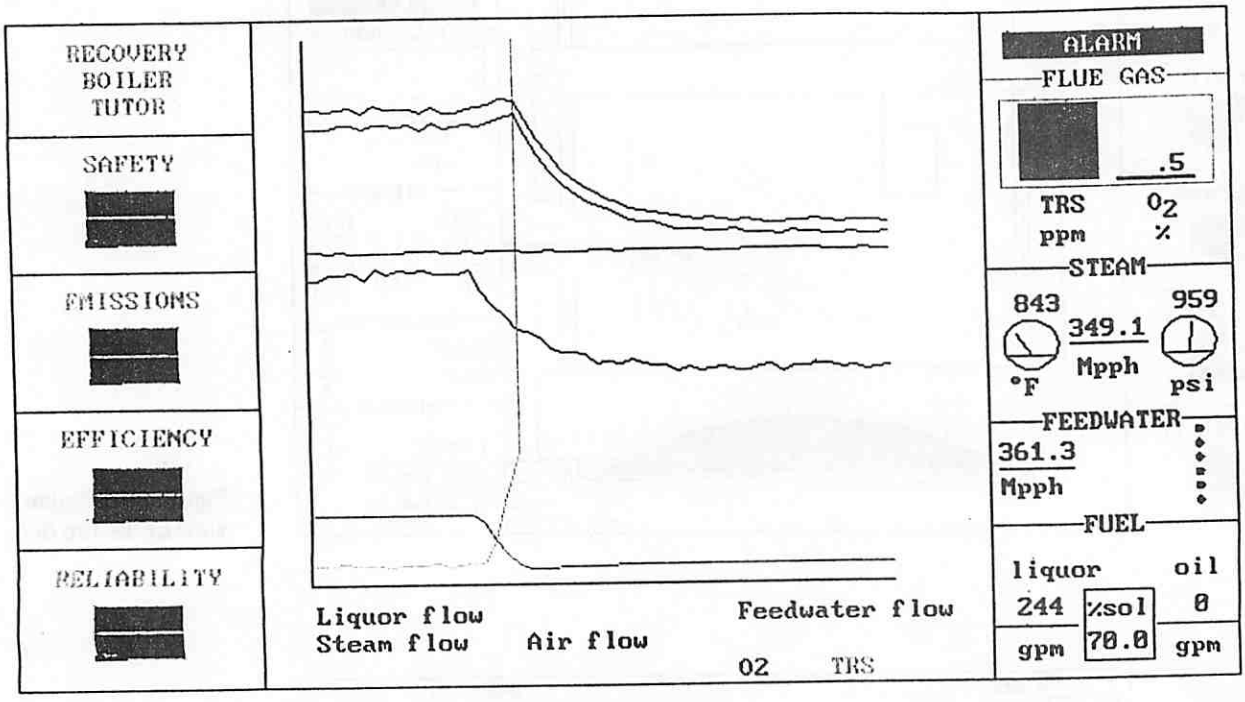

Figure 5. Trends selected by the operator.

oped in C, with many Lisp features implemented in C such as functional calls within the parameters of C functions. Meter readings and student interactions in the simulation were transferred between Fortran and C through vectors passed between the two programs.

Caleb for teaching a second language. Our second intelligent tutor teaches languages based on a powerful pedagogy called the "Silent Way"—a method developed by Caleb Gattegno that uses nonverbal communication within a controlled and artificial environment to teach second languages.[8] Learning a "second" language differs from acquiring a "first" language (as a child does). Children learning a first language must acquire linguistic competence such as the ability to associate meaning with sound, to make transformations, and to derive rules from observations and experimentation. Persons who have learned a first language have this linguistic competence and the Silent Way capitalizes on their ability by providing a directed environment engaging them in new, but analogous, linguistic experiences. The directed discovery environment engages students in an active learning process.

In Figure 6a, the tutor presents a new piece—a rod located in the center box. The student responds by typing the word for the new piece at the cursor. In Figure 6b, the student invents a new phrase by combining old pieces with the new one. In Figure 6c, the tutor corrects a student who places the adjective before (rather than after) the noun.

The tutor teaches Spanish by using graphical representations of Cuisenaire rods (originally developed by Gattegno for teaching arithmetic) to generate linguistic situations in which new words such as nouns, verbs, and adjectives are associated with meaning. On the screen, a rod is shown playing various roles; for example, it is used as an object to be given or taken by a student, or it is used to brush teeth. As a new rod is presented, students theorize about what situation is being represented, offer their conjectures, and revise their hypotheses as the situation demands. Students respond by typing a phrase to describe the situation in the text window. If the picture of a rod appears while the words *una regleta* are displayed, students type *una regleta* (as seen in Figure 6a). If students theorize that a new word such as *blanca* describes the size of the rod, they can later change that definition if (in fact) the new word defines the color of the rod (see Figure 6b). Meanwhile, students will have learned to write the word, spell it, and place it correctly in a sentence (see Figure 6c). They will have classified the word as a descriptor and will have invented phrases using it. Making and correcting hypotheses is central to language learning. Learner refinement of word meaning (that is, by a closer and closer approximation of expert ability) is one way to achieve linguistic mastery.

The tutor does not display words except to mention once, and only once, each new word in the target language. The tutor provides minimal *pieces* of the new language where *piece* is defined to be a phoneme, syllable, word, or phrase (see Figure 7). Pieces are aspects of the language that students can't invent, such as vocabulary and pronunciation. The tutor communicates silently, using icons, edit signals, and the rods—

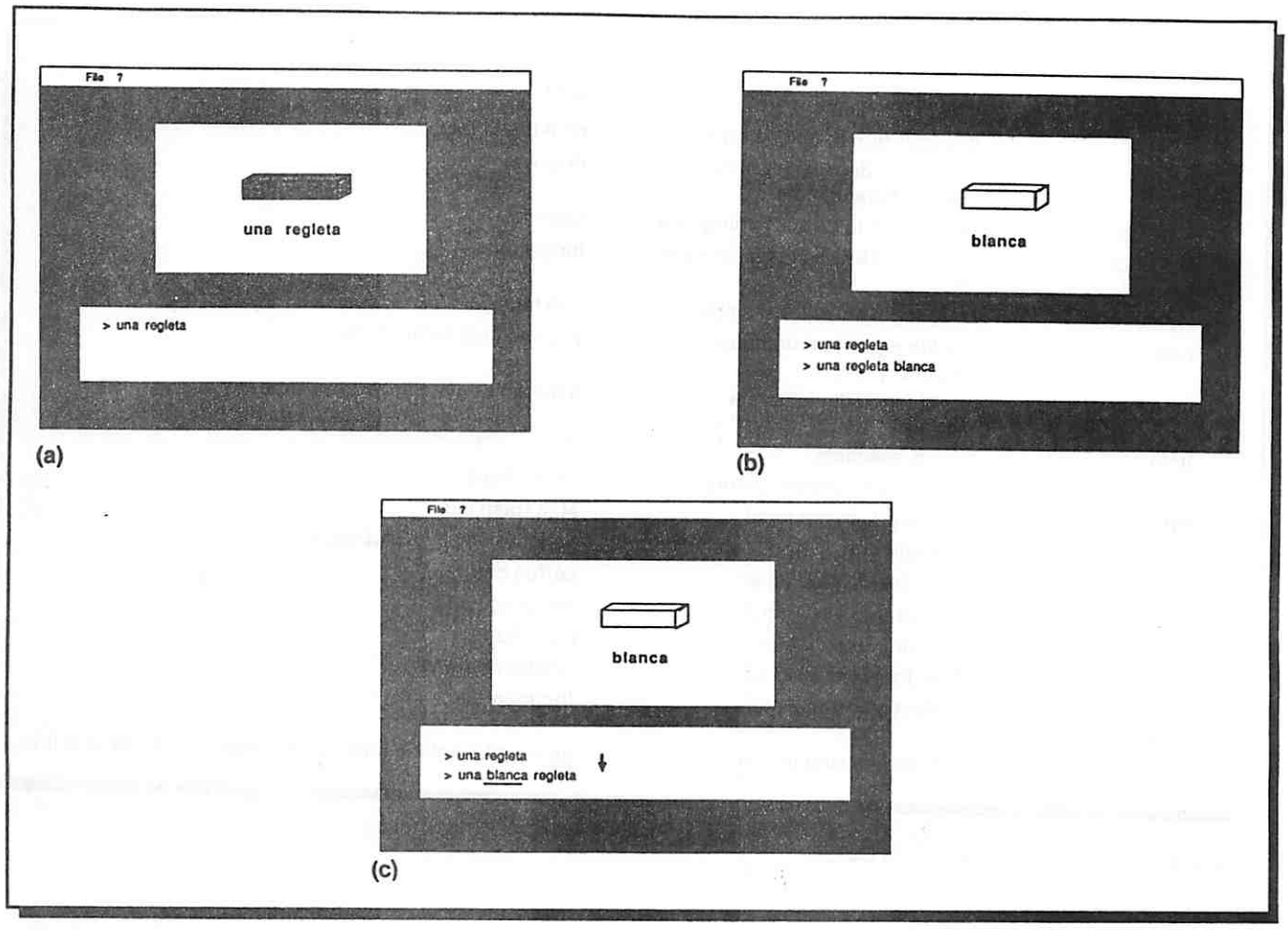

Figure 6. A system for teaching second languages.

Theme	Pieces	Potential misconceptions
1. Noun	a rod, una regleta	word order/agreement
example response: una regleta		
2. Adj	white, blanca grey, gris striped, listada dotted, punteada	word order/agreement
example response: una regleta blanca		
3. Conj	and, y	word order/use in series
example response: una regleta gris y una regleta blanca — una regleta gris, una regleta blanca, y una regleta negra		
4. Numbers	two, dos -s three, tres -s one, una	word order/agreement
example response: dos regletas blancas y tres regletas negras		
5. Noun deletion	one, blank ones	use in series/agreement
example response: una regleta roja y una blanca - dos regletas blancas y tres negras		
6. Verb + direct object	take, toma	word order/agreement
example response: toma una regleta gris - toma una regleta gris y tres blancas		
7. Verb + indirect object	give me, dame	word order/pronoun/agreement/case
example response: dame una regleta blanca - dame tres regletas negras y dos blancas		
8. Pronoun	it/them, la/las	word order/pronoun/agreement/case
example response: toma una regleta blanca y damela - toma tres regletas negras y damelas		

Figure 7. Themes and pieces in the Spanish curriculum.

Label	Idea to get across	Icon
go	it is your turn to do something	blinking cursor
wait	tutor busy, don't worry about nothing happening	Mac watch
attention	tutor about to do something new	sound
signal OK	let student know his response is OK	happy face
signal error	let student know he has error	sad face/Mr. Yuk
puzzled/repeat	say/do again (unintelligible response)	puzzled face/again sign
more	say/do more (incomplete response)	hand pull
help	help is available	?
dictionary	list of words already covered	book shape
throw out	extra stuff, do not need	Mac trash can
stop	save and quit = good bye	hand waving bye/stop sign
pause	take a break/stop timer	coffee cup
take	icon for mouse action	grasping hand
give	icon for mouse action	open hand
pacing regulator	slow down or speed up	speedometer
frustration gauge	student emotional state	thermometer
error correcting	word processing techniques	highlight/blinking cursor/placement arrows/fade in line

Figure 8. Communication icons in Caleb.

only "speaking" to provide words that students have not yet heard. Figure 8 lists icons used to represent these gestures, edit signals, and pantomime. These icons are used by the tutor, who plays the role of an orchestrator and monitor rather than that of an information giver.

With the introduction of verbs, action becomes possible. For example, the tutor prompts for a command by indicating a hand taking two white rods in the graphics window. The student types the command *Toma dos regletas blancas* ("Take two white rods") and the tutor performs the action.

So that students both produce and understand language, the tutor triggers two kinds of response: word-oriented responses typed at the keyboard, and action-oriented responses performed with the mouse and pictured objects. An example of the latter is the tutor giving the command *Toma dos regletas blancas* ("Take two white rods"). Students respond by using the mouse to take two pictured white rods with a grasping-hand-shaped cursor.

The non-verbal communication (that is, pantomime such as gestures, nods, and hand signals) of a Silent Way tutor are used to teach students to recognize when it is their turn to produce a sentence, when the tutor is about to say something, when the tutor expects more than students have produced, when an error needs correcting, or how to get help when they are stuck.

Errors are produced as students test and revise their theories about the language. When an error does occur, students are not deluged with entire sentences, nor are they provided a correct model to imitate. Instead, the precise location of the error is pointed out, as in Figure 6c, so that students may correct it themselves. The goal is to to let students develop their own sense of correctness or *inner criteria* for the new language.

The tutor monitors student input for correctness. A fault-tolerant parser filters "noisy" input so that some errors are ignored and some are treated, depending upon the situation. When the tutor treats an error, only the piece, noun, verb, or adjective that needs correcting is pointed out and students edit their input.

Regarding the presentation order of new material, the tutor makes decisions based on whether it's teaching the current piece, old piece, or old theme (see Figure 7). It uses five contexts to determine the number of times students should practice the piece. In the *intro context,* for example, the tutor simply presents the first example on the list of examples associated with each piece. When the tutor is in the *practice context,* the example source remains the same and the tutor marches down the list in a fixed order. In the *more practice context,* examples are chosen randomly from the piece example list. In the *review context,* examples are taken from an example pool of either the current theme or old themes. In the *error context,* examples come from the list associated with the error itself.

This system is implemented on a Macintosh and a Sun.

ESE for teaching thermodynamics. A third tutor built by Exploring Systems Earth—three universities working together to develop intelligent tutors—ESE is now in the early stages of implementation. This third tutor is one of a set of tutors that use interactive and monitored simulations for teaching elementary physics at high school and college levels. The goal is to put students in direct contact with physics elements such as mass, acceleration, and force. Students pursue various activities, such as changing the position or velocity of bodies in a celestial mechanics simulation, while viewing dependent changes in the size, speed, and position of orbit to improve their intuition about physical concepts. Each tutor monitors and advises students while providing examples, analogies, or explanations based on student actions, questions, or responses.

The tutor we are working on addresses the second law of thermodynamics—the law stating that heat cannot be absorbed from a reservoir and completely converted into mechanical work. This law is taught at the atomic level using a rich environment through which the principles of equilibrium, entropy, and thermal diffusion can be observed and tested.[9] Students are shown (and are able to construct) collections of atoms that transfer heat to one other through random collision (see Figure 9).

Students can create areas of high-energy atoms, indicated by dark squares, along with variously shaped regions within which the system can be analyzed and monitored. Several systems can be constructed, each with specific areas of high energy and associated observational regions. Concepts such as temperature, energy density, and thermal equilibrium for each observational region can be plotted against each other and against time. Students can then observe thermodynamic principles, such as heat transference through random collision and entropy as a function of initial system organization.

Students can modify system temperature, the number of collisions per unit time, and the shape of the initial "hot" region at any time. Changes in these parameters will cause dependent changes in the system. The tutor uses all student activities—including questions, responses, and requests—to formulate its next teaching goal and activity. It uses student actions to determine whether to show an extreme or near-miss example, whether to give an analogy or to ask a question. It bases its lessons on an ordered sequence of topics. To refine the tutor's response, we are now studying student misconceptions and common errors in learning thermodynamics and statistics.

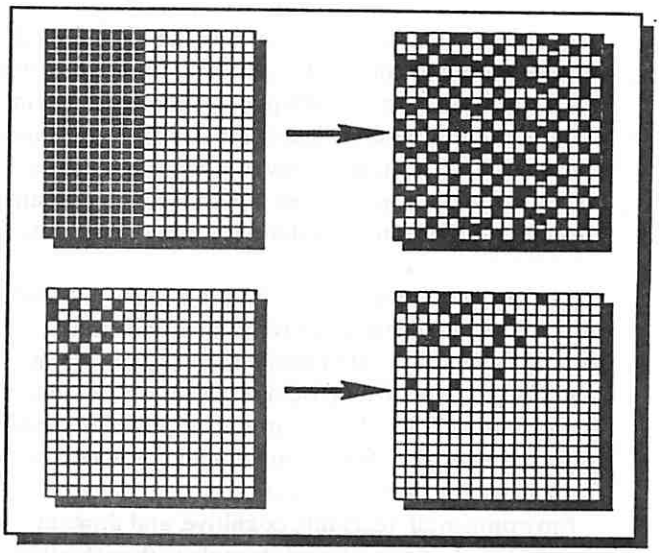

Figure 9. Systems moving towards equilibrium.

Klaus Schultz is ESE's domain expert, and Tom Murray the domain and teaching expert. Craig Lant, Paul Duquette, and Miguel de Campos are the environmental experts. The three universities comprising Exploring Systems Earth are the University of Massachusetts, San Francisco State University, and the University of Hawaii.

The need for multiple experts

Observations about the participation of experts in building these and other intelligent tutoring systems suggest that multiple experts must work on system design and implementation.[1,10] Much knowledge is distributed, subjective, unorganized, and misunderstood. No one human can supply the knowledge required—knowledge including environmental, teaching, cognitive, and domain expertise.

Given the complex and heterogeneous nature of knowledge required, we need methodologies and tools to transfer teaching and learning knowledge from each expert to the system under construction. Currently, few such tools exist.

Expert system shells contain a framework for building knowledge bases about concepts and rules, and for making inferences about them. Shells make building a tutor simpler to write.[11,12] However, we need more specific tools for designing and storing tutoring knowledge, and shells are limited in this respect. They are

frequently based on production rules and are limited in representing history and dependency of the tutoring interaction. Also, they inadequately represent tutoring and misconception knowledge—such as how to reason about teaching strategies, how to update and assess student models, how to select a path through domain concepts, and how to remediate for misconceptions.

The environmental expert. The first expert needed to build an intelligent tutor is the environmental expert. This person often uses a majority of system memory[1] to provide an envelope within which students and system interact. The environment provides specific tools and operators for solving domain problems or for performing domain activities.

Environmental, teaching, cognitive, and domain expert contributions interact strongly with each other—especially that from the environmental expert. For example, a system asking students to record entrance and exit angles for light rays in an optics experiment implies that the environment supplies such measuring devices.

The following criteria for developing tutoring environments have begun to emerge:

(1) Environments should be intuitive, obvious, and fun. Student energy should be spent learning the material, not learning how to use the environment.[13] For example, to indicate errors, express feelings, or convey meaning, the second language tutor's visual activities (see Figure 6) mimic the human Silent Way teacher's gestures, facial expressions, and rods. Each icon is designed to be clear, unambiguous, and to make use of student intelligence, experience, and resourcefulness.

(2) Environments should record not only what students do but what they intended to do, might have forgotten to do, or were unable to do.[14] Environments should provide a "wide bandwidth" within which multiple student activities can be entered and analyzed. For example, the Pascal tutor developed by Johnson and Soloway processed and analyzed an entire student program before offering advice.[15]

(3) Environments should be motivated by teaching and cognitive knowledge about how experts perform tasks and the nature of those tasks. For example, Anderson performed extensive research with geometry students before developing his geometry tutor interface,[16] and Woolf et al. incorporated knowledge from experts with more than 30 years experience working with boiler operations before building the RBT interface.[7]

(4) Environments should isolate key "tools" for attaining expertise in the domain. The economics tutor provided and monitored student use of key tools for performing economic experiments.[17] The RBT tutor provided process parameter graphs tracing trends over time, and used abstract meters to help operators reason about complex processes—thereby enabling them to make inferences about the effect of their actions (see Figures 2 through 5).

(5) Environments must maintain physical fidelity: Fidelity measures how closely simulated environments match the real world.[18] High fidelity identifies a system as almost indistinguishable from the real world. The RBT tutor presents a mathematically exact duplicate of the industrial process. It models and updates over 100 parameters every two seconds. Visual components of the industrial process such as alarm boards, control panels, dials, and reports are duplicated from the actual control room.

(6) Environments should be responsive, permissive, and consistent.[19] They should target applications based on skills people already have (such as moving icons) rather than forcing people to learn new skills. By responsive, we mean that student actions should have direct results—that students need not perform rigid sets of actions in rigid and unspecified order to achieve goals. By permissive, we mean that students may do anything reasonable and that multiple ways should exist for taking action. By consistent, we mean that moving from one application to another (for example, from editing text to developing graphics) should not require learning new interfaces. All tools should be based on similar interface devices, such as pull-down menus or single and double mouse clicks.

No environment is appropriate for every domain: We must study each domain to determine how experts function in that domain, how novices might behave differently,[20,21] and how novices can be helped to attain expertise.

The teaching expert. Acquiring sufficient and correct teaching expertise is a long-term problem for builders of tutoring systems—in part because sophisticated knowledge about learning, teaching, and domain knowledge remain active areas of research in most domains. For the machine tutor, designing decision logic and rules to guide tutorial interaction is a process of successive iteration—a process that can be improved continuously.

Tools to facilitate inclusion of long-term teaching knowledge are just beginning to emerge. For example,

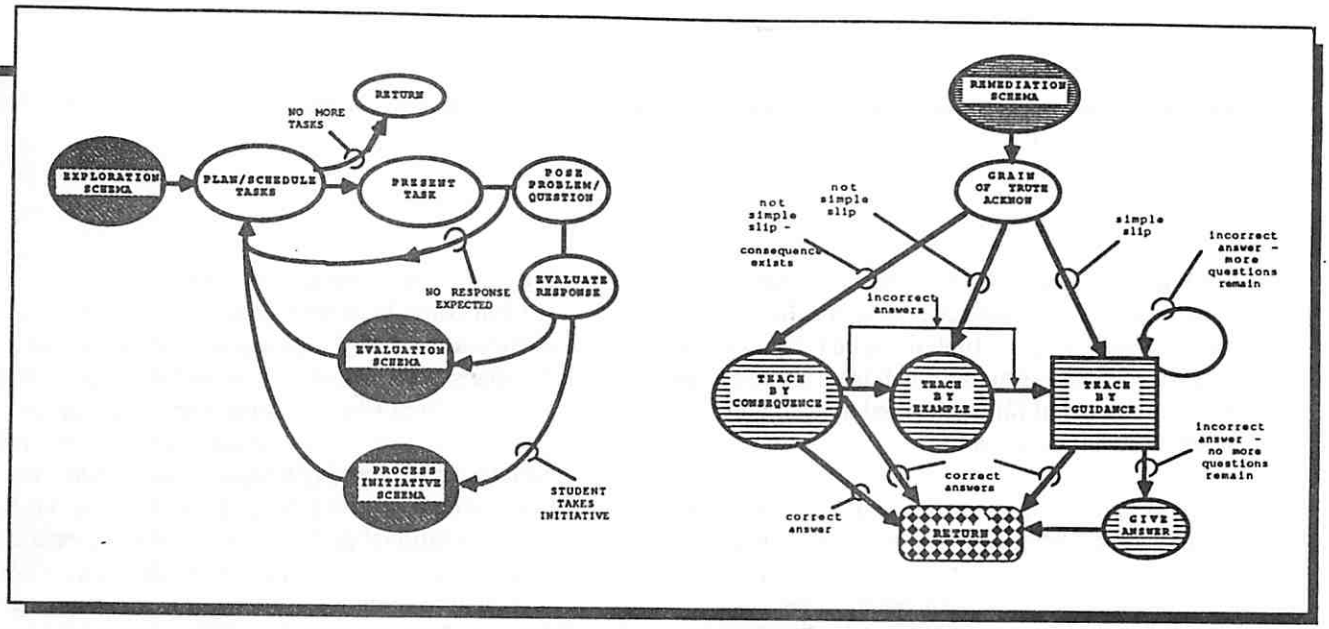

Figure 10. A framework for managing tutoring discourse.

we have developed a framework for managing discourse in an intelligent tutor that reasons dynamically about discourse, student response, and tutor moves (see Figure 10).[22] This is a flexible and domain-independent framework that is portable to various machine tutors. Also, it can be rebuilt—decision points and machine actions are modifiable for fine-tuning system response.

The framework reasons about which pedagogical responses to produce and which alternative discourse moves to make. It custom-tailors feedback to students in the form of examples, analogies, and simulations. Discourse *schemas* have been defined as collections of discourse activities and response profiles. Discourse is planned by passage through the schemas. The number and type of schemas selected depend on context.

We used empirical criteria to define these schemas: Tutoring responses were analyzed from empirical studies of teaching and learning,[23] other responses appeared as effective teaching strategies in research on misconceptions in physics,[24,25] others obeyed felicity laws, [26] and still others obeyed general rules of discourse structure.[27,28] General rules and fundamental remediation strategies have been incorporated into our discourse schemas and included in our teaching framework.

By continuously adjusting to real-time changes in either the knowledge base or the user, the discourse framework allows the tutor to remain flexible while cooperatively engaged in conversation. The framework views discourse as navigation through a set of possible discourse situations (see Figure 10). We are now using this framework to improve the physics tutor's response to idiosyncratic student behavior. The structure is preliminary in the sense that it's designed to be rebuilt; response decisions and machine actions, explicitly represented in the system, can be modified through a graphics editor. Appropriate machine response can be assessed and continuously improved through the editor. In the long term, we intend to make this reasoning process available to human teachers, who can then modify the tutor for use in a classroom.

No single teaching strategy is appropriate for every domain: Each domain and each student must be assessed to determine an appropriate teaching strategy. For example, Anderson et al. built geometry and Lisp tutors that respond immediately to incorrect student answers.[11] These authors argued that they needed immediate computer feedback to avoid fruitless student effort. They suggested that erroneous solution paths in geometry and Lisp might be so ambiguous and delayed that errors would not be recognized by students if tutor therapy were delayed.

This pedagogy is opposite to that used by Cunningham et al.[8] and Woolf et al.[7] These tutors were passive (not intrusive) advisors. Their strategy was to subordinate teaching to learning, allowing students to experiment while developing hypotheses about the domain. They guided students toward developing student intuitions, but did not correct students so long as student performance appeared to be attaining a precise goal.

In industrial settings, particularly, trainees must learn to generate multiple hypotheses and to evaluate their performance based on how their actions affect the industrial process. No human tutor is available during normal boiler operation. Thus, the machine's teaching strategy encouraged students to trust their own observations about the industrial process—

helping them to learn through animated simulations, trend analyses, and real-time, dynamically updated meters. In addition, textual dialogue (1) assured that operators would extract as much information as possible from data, and (2) established a mechanism to redirect them if they did not.

The cognitive expert. At present, the role of the cognitive scientist is incompletely understood; in part, this researcher seeks to discover how people learn and teach in a given domain. For example, cognitive science research in thermodynamics will enable systems to recognize common errors, tease apart probable misconceptions, and provide effective remediation. Cognitive science research provides the tutor with a basis for selecting instructional strategies. The importance of addressing common errors and misconceptions in physics is well documented, and the tutor's intelligence hinges on making that knowledge explicit.[24,25]

We want a tutoring system to help students (1) to generate those hypotheses that are necessary precursors to expanding their intuition, and (2) to develop their own models of the physical world, while discovering and "listening to" their own scientific intuitions. To do this, we rely on work done by cognitive scientists, who study how students reason about qualitative processes, how teachers impart propaedeutic principles (or the knowledge needed for learning some art or science),[29] and what tools are being used by experts working in the field.

For example, the cognitive science experiments that must be performed to build our thermodynamics tutor include (1) investigation of real-world tools currently used by physicists, (2) examination of studies that focus on cognitive processes used by novices and experts, and (3) comparison of novice with expert understanding of physics problems. Cognitive science can define such knowledge—knowledge that elucidates actions taken by experts to make measurements or perform transformations in a domain. We call this "heuristic knowledge" and define it as *knowledge about how to solve domain problems.* Heuristic knowledge differs from procedural knowledge in that it adds neither content nor concepts to the domain, but describes actions taken by experts using their conceptual and procedural knowledge. Such knowledge, rarely included in tutoring systems, must be included if tutors are to monitor student problem-solving activities and experiential knowledge about how to work in the field.

RBT articulates this knowledge by explicitly recording student attempts to solve emergencies. It shows students their false paths and gives reasons behind particular rule-of-thumb knowledge used to solve problems. RBT also provides students with various examples from which they can explore problem-solving activities—showing students their own paths, preferred paths, and (perhaps, in time) their own underlying cognitive processes. Simply elucidating these operational problem-solving components in a domain, and the rules applying to their use, is not sufficient to understand how one learns in a new domain. By using such knowledge, however, a tutor can help students *learn how to learn.* We are compiling such data and expect that it (along with cognitive studies) will elucidate some processes behind problem-solving behavior.

The domain expert. An in-house domain expert is a critical requirement for building intelligent tutoring systems. By "in-house," we mean that the domain expert must join the project team for anywhere from six months to several years while domain knowledge is being acquired. Less commitment than that—that is, any role less than that of full-fledged team member—suggests a less-than-adequate transfer of domain knowledge.

In the tutors described above, domain experts were (and are) integral to the programming effort. The programmer, project manager, and director of RBT were themselves chemical engineers. More than 30 years of theoretical and practical knowledge about boiler design and teaching strategies were incorporated into the system. Development time for this project would have been much longer than 18 months if these experts had not previously identified the boiler's chemical, physical, and thermodynamic characteristics and collected examples of successful teaching activities.

An instructor holding an ESL graduate degree (English as a second language) developed the second-language tutor. This instructor (the second author of this article) has more than seven years teaching experience. She has used the Silent Way to teach intensive English courses to foreigners living in America, and to train Nepali language instructors who in turn taught Nepali to future American Peace Corps volunteers.

The physics tutor is being built after more than 18 months of work with member physicists and astronomers. In addition, potential users of the system (high school and college physics teachers) are contributing environmental and teaching knowledge. Their overall effort produced more than 100 pages of rules, processes, screen designs (including help activi-

ties about physics), and cognitive studies (identifying educational goals, potential errors and misconceptions) before any code was built.

For domain knowledge, expert shells can be used effectively. Anderson[11] used Grapes[30] to represent the rules programmers use for solving problems, to describe Lisp functions, and to represent higher level programming goals. He used *buggy* rules to represent misconceptions that novice programmers often develop during learning. Streibel et al.[12] used OPS5 to write rules for genetic problem solving and to encode teaching strategies.

Based on the various expert systems that have been built, the following criteria for acquiring domain knowledge are well understood:

(1) Domain experts should be true experts—if possible, the best in the field.[1] Dendral, for example, an expert system for generating and testing hypotheses about chemical structures and spectroscopic data, was built by a team including Joshua Lederberg (a Nobel-prize-winning geneticist), Carl Djerassi (a world-class expert on mass spectral analysis), and other professional chemists and computer scientists.[31]

(2) Domain experts are expensive. Gaining the attention of knowledgeable people in any domain is expensive and time consuming. However, the willingness and availability of such experts to participate is critical to the knowledge-engineering process. Assigning the expert role to someone of lesser ability (or worse, to persons with "time on their hands") might doom a project to failure. On the other hand, enthusiastic support from funders and supervisors—including sufficient allocation of resources, human and otherwise—are prerequisite to success.

(3) Certainly, individual domain experts can have incomplete knowledge or conceptual vacuums. Multiple experts are needed for testing and modifying domain knowledge throughout the tutor's life.

(4) Similarly, domain knowledge can be overly distributed.[1,10] That is, knowledge can be spread so diffusely among different research projects and experts as to leave any system unfinishable that uses only a single expert (or even several experts). Thus domain knowledge must be acquired incrementally and must be prototyped, refined, augmented, and reimplemented. The time needed to build a tutoring system "should be measured in years, not months, and in tens of worker years, not worker months."[1]

(5) Domain knowledge found in textbooks is incomplete and idealized.[1] In an intelligent tutoring system, textbooks are inappropriate as a primary source for either domain or teaching knowledge. Textbooks rarely contain the commonsense knowledge—the know-how used by expert tutors or professionals in the field—to help choose a next-teaching strategy or solve difficult problems. Books tend to present clean, uncomplicated concepts and results. To teach or solve real-world problems, tutors must know messy but necessary details of real and perceived links between concepts and unpublished rules of teaching and learning.

Intelligent tutors can neither cure all educational problems nor totally answer the dilemma facing education. They seem incapable of achieving the very difficult, let alone the impossible. However, they do provide exciting possibilities—and of these, one of the most exciting is that of providing a community memory for teaching and learning research. This community memory would provide a focus for articulating distributed knowledge in an intelligent tutor. It would include recent as well as historical research about thinking, teaching, and learning. Evaluating such an articulation would, in itself, contribute to education—and ultimately, to communication between experts.

Compiling diverse research results from environmental, teaching, cognitive, and domain experts is currently hampered by the lack of explicit methodologies and technologies to help authors transfer their knowledge to a system. This article has specifically addressed the issues of what further knowledge we need and where we should apply human and financial resources to build future intelligent tutoring systems. ■

Acknowledgments

This work was supported in part by the Air Force Systems Command, Rome Air Development Center, Griffiss AFB, New York—and by the Air Force Office of Scientific Research, Bolling AFB, the District of Columbia—under contract No. F30602-85-C-0008. This contract supports the Northeast Artificial Intelligence Consortium.

References

1. D. Bobrow, S. Mittal, and M. Stefik, "Expert Systems: Perils and Promise," *Comm. ACM*, Sept. 1986.
2. J. McDermott, "A Rule-Based Configurer of Computer Systems," *Artificial Intelligence*, Jan. 1982.
3. R. Smith, "On the Development of Commercial Expert Systems," *AI Magazine*, Fall 1984.
4. D. Waterman, *Innovation: The Attacker's Advantage*, Simon & Schuster, New York, N.Y., 1986.
5. D. Bitzer, D. Braunfeld, and W. Lichtenberger, "PLATO: An Automatic Teaching Device," *IRE Trans. Education*, Dec. 1961, pp. 157-161.
6. A. Bork, "Physics in the Irvine Educational Technology Center," *Computers in Education*, No. 4, 1980, pp. 37-57.
7. B. Woolf et al., "Teaching a Complex Industrial Process," *Proc. Am. Assoc. Artificial Intelligence*, Philadelphia, Pa., Aug. 1986.
8. P. Cunningham, T. Iberall, and B. Woolf, "Caleb: An Intelligent Second Language Tutor," *Proc. IEEE Int'l Conf. Systems, Man, and Cybernetics*, Computer Soc. IEEE, Los Alamitos, Calif., 1986, pp. 1210-1215.
9. T. Atkins, *The Second Law*, Freedman, San Francisco, Calif., 1982.
10. S. Mittal and C. Dym, "Knowledge Acquisition from Multiple Experts," *AI Magazine*, Summer 1985.
11. J. Anderson, C. Boyle, and G. Yost, "The Geometry Tutor," *Proc. Ninth Int'l Joint Conf. Artificial Intelligence*, Los Angeles, Calif., Aug. 1985.
12. M. Streibel et al., "An Intelligent Computer Tutoring System for Solving Genetics Problem Solving, Conjecturing, and Understanding," *Machine-Mediated Learning*, Vol. 2, Nos. 1 and 2, 1987, pp 129-159.
13. P. Cunningham, *Caleb: A Silent Second-Language Tutor: The Knowledge Acquisition Phase*, Tech. Report #87-6, Rensselaer Polytechnic Institute, Troy, N.Y., 1986.
14. R. Burton, "Instructional Environments," in *Foundations of Instructional Tutoring Systems*, M. Polson and J. Richardson, eds., Lawrence Earlbaum Associates, Hillsdale, N.J., in press.
15. L. Johnson and E. Soloway, "Intention-Based Diagnosis of Programming Errors," *Proc. Am. Assoc. Artificial Intelligence*, Austin, Tex., Aug. 1984.
16. J. Anderson, "Tuning of Search of the Problem Space for Geometry Proofs," *Int'l Joint Conf. Artificial Intelligence*, British Columbia, Canada, Aug. 1981.
17. V. Shute and J. Bonar, "Intelligent Tutoring Systems for Scientific Inquiry," *Proc. Cognitive Science Soc.*, Lawrence Earlbaum Associates, Hillsdale, N.J., 1986.
18. J. Hollan, E. Hutchins, and L. Weitzman, "Steamer: An Interactive Inspectable Simulation-based Training System," *AI Magazine*, Summer 1984.
19. Apple Corp., *Inside Macintosh*, Vol. 1, Addison-Wesley, Reading, Mass., 1985.
20. W. Chase and H. Simon, "The Mind's Eye in Chess," in *Visual Information Processing*, W.G. Chase, ed., Academic Press, New York, N.Y., 1973.
21. J. Larkin et al., "Expert and Novice Performance in Solving Physics Problems, *Science*, June 1980.
22. B. Woolf and T. Murray, "A Framework for Representing Tutorial Discourse," to appear in *Proc. Int'l Joint Conf. Artificial Intelligence*, Milan, Italy, Aug. 1987.
23. J. Clement and D. Brown, "Using Analogical Reasoning to Deal with Deep Misconceptions in Physics," tech. report, Cognitive Research Group, Physics Department, University of Mass., May 1984.
24. J. Clement, "Students' Preconceptions in Introductory Mechanics," *Am. J. Physics*, Jan. 1982.
25. L. McDermott, "Misconceptions in Classical Mechanics," *Physics Today*, July 1984.
26. K. van Lehn, "Felicity Conditions for Human Skill Acquisition: Validating an AI Theory," research report, Xerox Palo Alto Research Center, Palo Alto, Calif., 1983.
27. B. Grosz and C. Sidner, "The Structure of Discourse Structure," *Proc. Am. Assoc. Artificial Intelligence*, Los Angeles, Calif., Aug. 1985.
28. R. Reichman-Adar, *Plain Speaking: A Theory and Grammar of Spontaneous Discourse*, PhD dissertation, Harvard University, Department of Mathematics, Cambridge, Mass., 1981.
29. H. Halff, "Curriculum and Instruction in Automated Tutors," in *Foundations of Instructional Tutoring Systems*, M. Polson and J. Richardson, eds., Lawrence Earlbaum Associates, Hillsdale, N.J., in press.
30. R. Sauers and R. Farrell, "Grapes User's Manual," tech. report, Office of Naval Research 82-3, Carnegie Mellon University, Pittsburgh, Pa., 1982.
31. R.K. Lindsay et al., *Applications of Artificial Intelligence for Organic Chemistry*, William Kaufmann, Los Altos, Calif., 1985.

Beverly Woolf is an assistant professor of computer and information sciences at the University of Massachusetts. Her research interests include AI, cognitive science, knowledge representation, and human-computer interfaces for explanation and teaching. She and her research group have active projects in machine tutoring, student modeling, discourse management, and intelligent interfaces. She received her BA in physics from Smith College, and her MS and PhD in computer science from the University of Massachusetts.

Pat Cunningham is a systems programmer in the electrical engineering department at Yale University. She has taught mathematics, science, and English as a second language in Nepal and in the US. Her current research focuses on using computers in educational settings, especially intelligent tutoring systems. She received her BA in Spanish from Thiel College, her MA in ESL from the University of Hawaii, and her MS in computer science from Rensselaer's Hartford Graduate Center.

The authors can be reached in care of Beverly Woolf, at the Dept. of Computer and Information Science, University of Massachusetts, Amherst, MA 01003.