Intelligent Computer-Aided Training Authoring Environment

Robert D. Way
way@gothamcity.jsc.nasa.gov
(713) 483-1899

LinCom Corporation
1020 Bay Area Blvd. Suite 200
Houston, TX 77058

ABSTRACT

Although there has been much research into intelligent tutoring systems (ITS), there are few authoring systems available that support ITS metaphors. Instructional developers are generally obliged to use tools designed for creating on-line books.

We are currently developing an authoring environment derived from NASA's research on intelligent computer-aided training (ICAT). The ICAT metaphor, currently in use at NASA, has proven effective in disciplines from satellite deployment to high school physics. This technique provides a personal trainer (PT) who instructs the student using a simulated work environment (SWE). The PT acts as a tutor, providing individualized instruction and assistance to each student. Teaching in an SWE allows the student to learn tasks by *doing* them, rather than by *reading about* them.

This authoring environment will expedite ICAT development by providing a tool set that guides the trainer modeling process. Additionally, this environment provides a vehicle for distributing NASA's ICAT technology to the private sector.

INTRODUCTION

"Industrial, business and commercial training accounts for about half of the total educational expenditure in the United States."
(Richardson 1988)

Education, training and re-training are frequently documented as expensive and inefficient by the media. Recent research in intelligent tutoring systems (ITS) and intelligent computer-aided instruction (ICAI) are often promoted as the remedy to educational problems.

Anecdotally, these claims seem well founded. Schank's "Case-Based Teaching" (Riesbeck 1991) and Woolf's "Discourse Management" (Woolf 1991) demonstrations both show amazing ability to intelligently interact with students. Unfortunately, the theories behind these systems are still the subject of intensive research. It may take several years before training based on these metaphors is commercially available. At any rate, many ITSs have been around long enough to show an influence on the commercial market. Equally as innovative, and more often quoted, Anderson's Geometry and LISP Tutors, (Anderson 1985) Johnson & Soloway's PROUST system for Pascal programming (Johnson 1984), and Hollan & Hutchins' industrial trainer Steamer (Hollan 1984), were all well published successes before 1988.

Yet despite being updated with the latest multi-media effects and hyper-text links, most available educational products do not employ any of the instructional techniques pioneered by ITSs during the past decade.

One reason for this lack of transfer is that while there has been much research into intelligent tutoring systems, there are few authoring systems available that support intelligent tutoring concepts. For example, no commercially available authoring tool supports all five common components (Burns 1991) of an ITS: domain expert, instructional expert, student model, intelligent interface, and simulation.

Instructional designers are commonly obliged to use general purpose authoring tools. These tools, like Authorware and ToolBook, shield designers from the complexities of color graphics, digitized sound and video, but do nothing to assist in the intricacies of ITS design. Deprived of basic student modeling capabilities, designers are discouraged from

creating systems which vary remediation based on the ability of each student. Worse yet, lacking new instructional metaphors, general purpose tools tend to promote systems which are either high-tech slide shows or on-line books.*

Intelligent Computer-Aided Training

We are developing an authoring environment derived from NASA's research on intelligent computer-aided training (ICAT). The ICAT instructional metaphor, currently in use at NASA, has proven effective in disciplines from satellite deployment to high school physics (see related work section). This technique provides a personal trainer (PT) who instructs the student within a simulated work environment (SWE). The PT acts as a personal tutor, providing individualized instruction and assistance to each student. Teaching in an SWE allows the student to learn tasks by *doing* them, rather than by *reading about* them.

Figure 1 shows a simulated environment developed to train astronauts in operating the Spacehab module. Concepts are presented in the personal trainer window on the right hand side of the screen. Students respond to the trainer by identifying and manipulating objects in the environment. Navigation around the shuttle is performed using the displays in the top right corner. Switches and dials are directly manipulated using the mouse.

Different fidelities of simulated environment and personal trainer are used depending on the needs of the training. In this case the personal trainer is implemented using only text. Other systems have required more personality. In these cases the PT is presented using video of a real person. This helps capture the feeling of interacting with a human trainer.

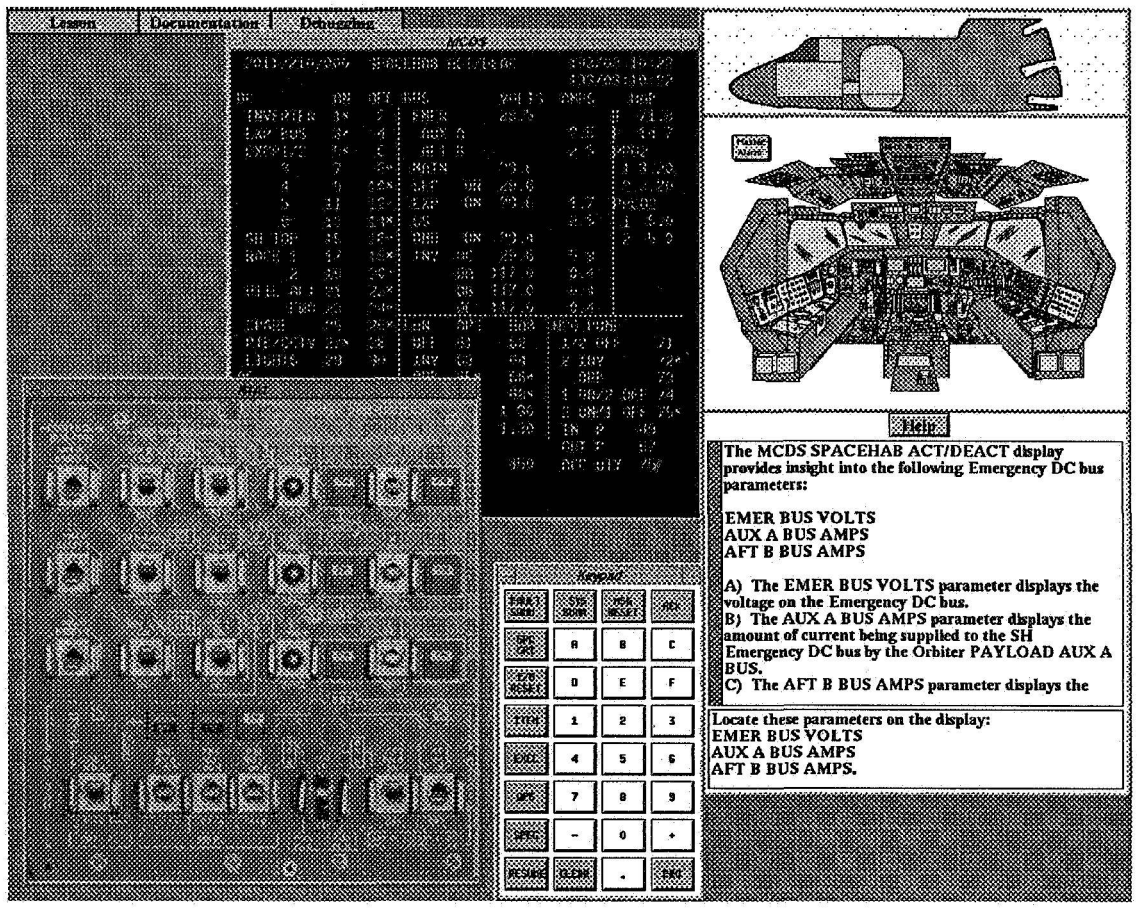


Figure 1
Simulated Environment for Space-Hab Training

* Capable instructional systems *have* been developed using current tools, but these systems owe more to the experience of the author than to the support of the tool.

ARCHITECTURE

The STB's authoring environment plans elaborate on NASA's second generation ICAT architecture (see Figure 2). NASA's architecture, shown in the lower level, defines which modules are necessary to build an ICAT, what each should do, and how they communicate. This architecture does not, however, specify *how* to implement each module. Previous ICAT system have used different methods of implementation. This project will provide a common library for all new ICAT systems and tools to guide the instructional designer through the development process.

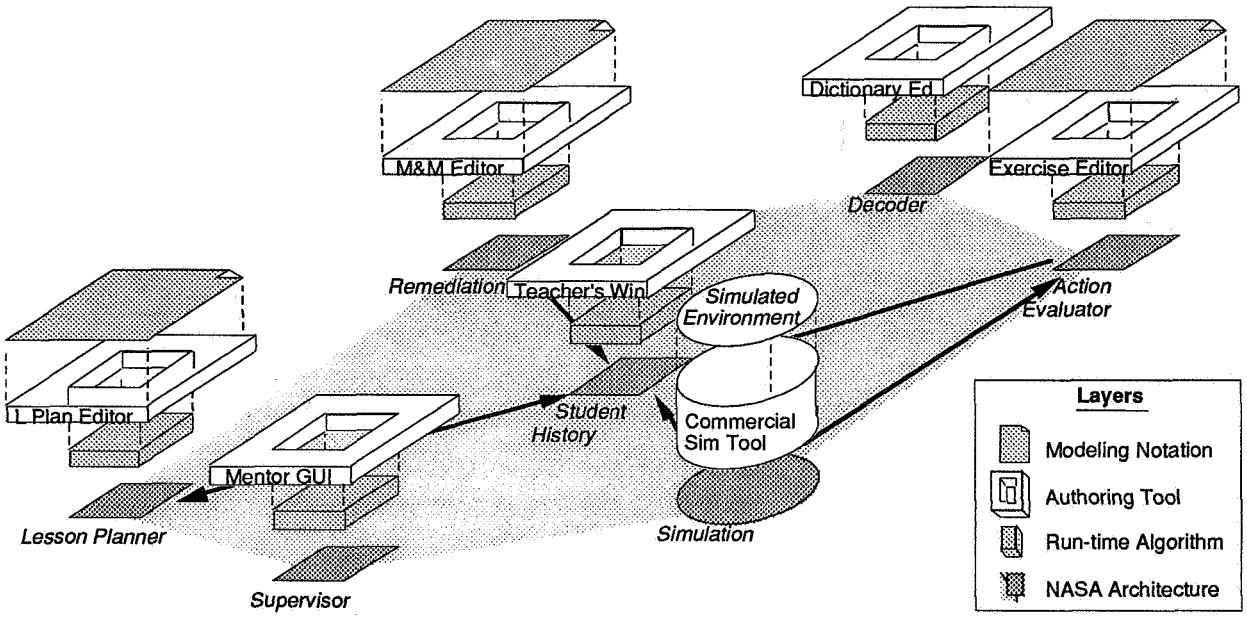


Figure 2
ICAT Architecture

Building an ICAT system based on this architecture requires both simulating the work environment and modeling a personal trainer. Unlike previous systems, Space station ICATs will use a pre-existing simulated work environment built for Part-Task Trainers. To simplify this process we have minimized the interaction between the SWE and the personal trainer modules of the ICAT system. The SWE only interacts with the PT through the student history and the action evaluator. The SWE is shown as the lone round module in Figure 2.

Rectangular modules in Figure 2 represent sections of the personal trainer. A personal trainer has three main duties, each assigned to a separate module:

Lesson Planner:	Assign appropriate exercises to each student.
Action Evaluator:	Watch what the student does and compare it objectively to the exercise's expected behavior.
Remediation:	Point out the student's mistakes and give assistance tailored the student's past performance.

The three other modules shown in Figure 2 provide basic support to these core routines. The **Student History** remembers what the student did, what the student knows, and what we told the student. The **Supervisor** provides the personal trainer's graphical interface with the student and coordinates execution of the ICAT. The **Decoder** acts as a mini database for the ICAT. It maintains a list of all actions, concepts, exercise names, messages, and misconceptions used in a particular ICAT. These support modules are well understood (for the purposes of this proposal) and are not discussed further.

After reviewing previous ICAT implementations and relevant literature, The STB has selected synergistic implementations for each ICAT module (Lesson Planner, Remediation, etc.). On top of each module, The STB has layered a graphical modeling notation. These notations enable an instructional designer to easily diagram a new ICAT design.

But, while these notations greatly aid in specifying ICAT knowledge, someone with programming knowledge of these particular algorithms must still hand translate the diagrams into computer data files. The main goal of this project is to construct an authoring environment which helps an instructional designer create a new ICAT system. This environment will provide graphical editors for each modeling notation, and automatically translate diagrams into the required data files. In addition, these editors will be integrated with the run-time algorithms. This will allow the editors to also act as debugging tools . While a student is working an exercise, the editors will automatically highlight the ICAT diagrams to show the student's progress.

The remainder of this document is broken into four sections, one explaining the SWE and three explaining to the core modules above.

Simulated Work Environment

Figure 3 shows one of the approximately twenty-five panels that make up the Spacehab Intelligent Facilities Trainer (SHIFT) Simulated Work Environment. Each of the switches, buttons and lights on the panel is active. An astronaut throws switches using the mouse. Indicator lights are controlled by an underlying engineering model of the Spacehab module.

Astronauts learn procedures which involve throwing a series of switches and checking for the appropriate indicators to light up. They also monitor the system, listening for alarms and looking for problem "signatures".

The GUI and engineering models of previous SWEs have required up to two-thirds of the total effort allotted to creating an ICAT system. By using pre-existing part task trainers we expect to dramatically reduce the development time of space station ICAT systems.

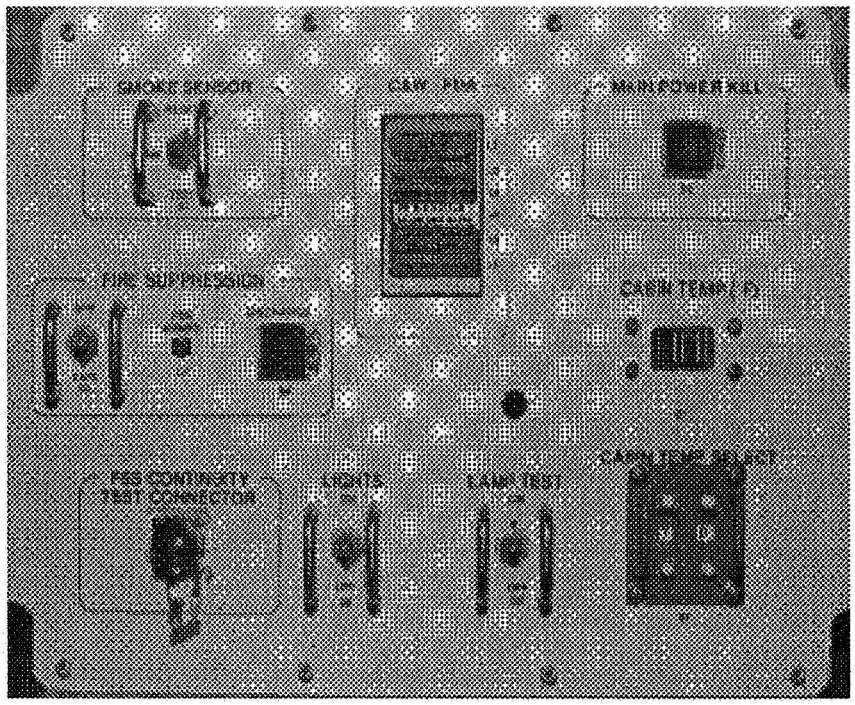


Figure 3
Panel from the SHIFT simulated work environment

Action Evaluator

Once an SWE is defined, the instructional designer must develop exercises to be performed in the environment. ICAT theory defines two different types of exercises: those which present new material to the student and those which apply the material. These concept application lessons also serve as the way the ICAT system evaluates the student's progress.

The STB has developed a graphical notation called operating procedure language (OPL) which enables an instructional designer to specify what things a student should do during a lesson. Figures 4 and 5 show examples of this notation. The STB has also implemented algorithms which allow the action evaluator to compare the student's behavior to the specified notation.

Figure 4 shows the OPL notation for an exercise which presents a new procedure to the student. Rectangles represent actions the student is expected to carry out. Rounded rectangles represent the sequence interactions given to the student. Circles represent commands to the simulated work environment. Notice that a presentation exercise normally has a linear structure.

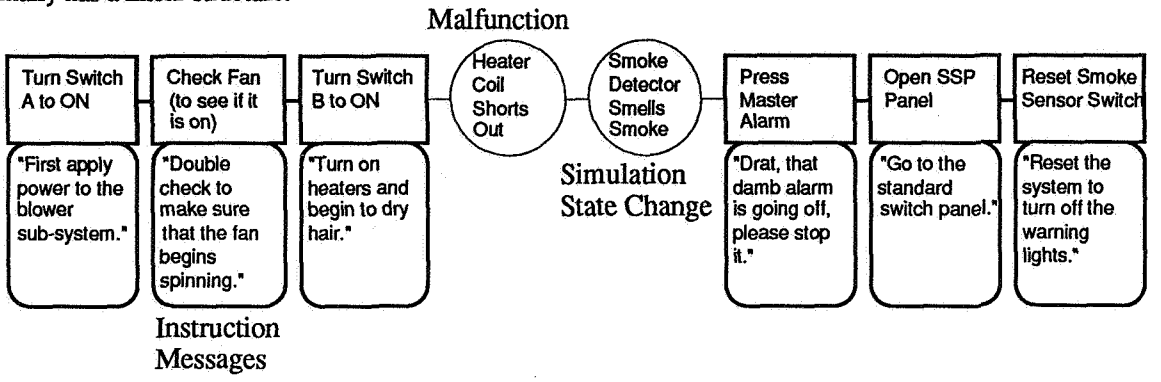


Figure 4
Material Presentation Exercise

Figure 5 shows an exercise which allows the student to practice a procedure. Practice exercises are not necessarily linear, they allow students to work a procedure using any equivalent series of steps. They also support branching of procedures based on the state of the SWE. Practice exercises may also be annotated to show common mistakes.

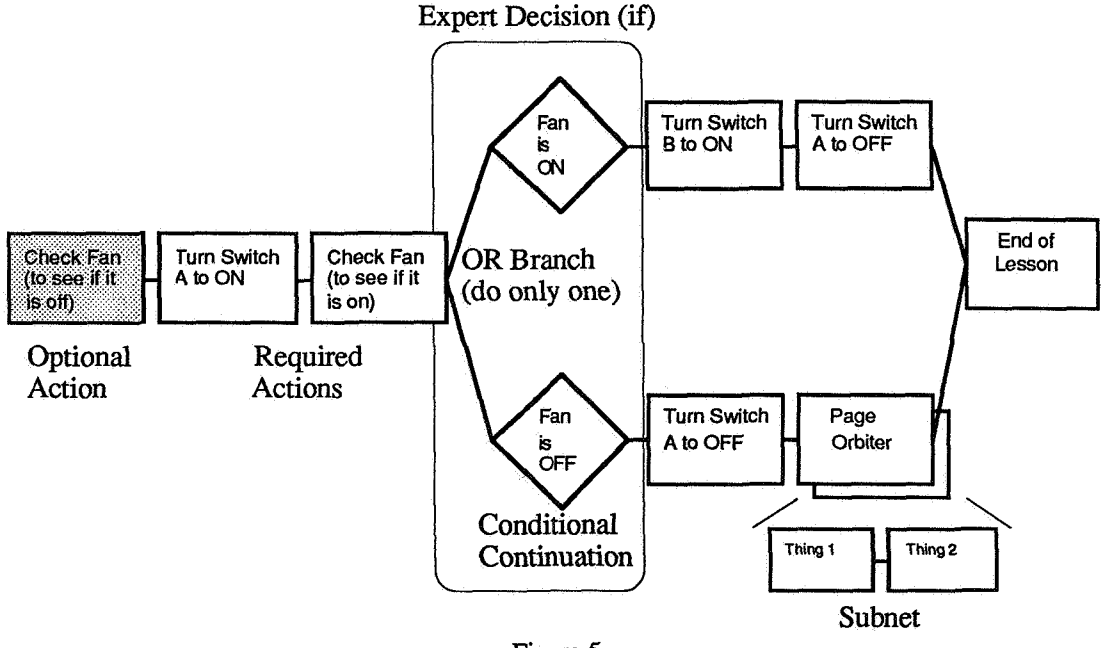


Figure 5
Practice/Evaluation Exercise

During the SHIFT project, OPL greatly simplified the creation of exercises. Although OPL provides an efficient way of specifying exercises, each exercise must be translated by hand into the data files required for implementation. The new exercise editor will facilitate both drawing OPL diagrams and generating the data files.

Remediation

In the course of working an exercise, the student will complete a number of actions. Each action, either correct or incorrect, gives information about the student's understanding of the material. The major part of trainer modeling is watching the series of actions that the student performs, and identifying patterns of behavior which signify a misunderstanding of the material. Once a misunderstanding is identified, the trainer gives the student an appropriate message.

Trainer modeling is based on the assumption that we have available a human "trainer" who understands the material and could teach it well in a one-on-one setting. If no one exists who could teach the proposed material we have found it extremely difficult to create an effective ICAT system. If the assumption is true, however, trainer modeling simply reduces to "doing what the human trainer would do."

In creating various ICAT systems, we have talked to many teachers and trainers. These experts have told us that each student comes into the training program with a personal conceptual model of the world. This conceptual model drives the student's assimilation of new material. If the student's conceptual model is correct relative to the material being taught, the assimilation is usually easy. If the student's conceptual model is wrong, however, it can often lead to misinterpretation of the material. These misinterpretations often cause the student to make mistakes during an exercise.

Fortunately, many students have similar backgrounds and, therefore, have similar conceptual models of the world. Because these similar models cause students to make similar mistakes, good teachers learn to quickly spot behavior patterns which indicate misinterpretations of the material. This is a principle called "Misconception Theory." (Way 1991) Misconception theory holds that teachers can easily describe what common mistakes students make, what misconceptions causes these mistakes, and how to remediate them.

The remediation module of the personal trainer implements this process. The STB has developed an efficient matching algorithm which stores patterns of behavior (student actions) and maps them to misconceptions. This algorithm is facilitated by imbedding the pattern matching in a binary search tree (see Figure 6). In addition, the algorithm provides for structured relationships between misconceptions. This allows the system to "fallback" and remediate a more general concept if several related misconceptions have been diagnosed (see Figure 7). Thirdly, it stores one or more types of remediation for each misconception. This allows the system to give appropriate levels of explanation based on the student's background.

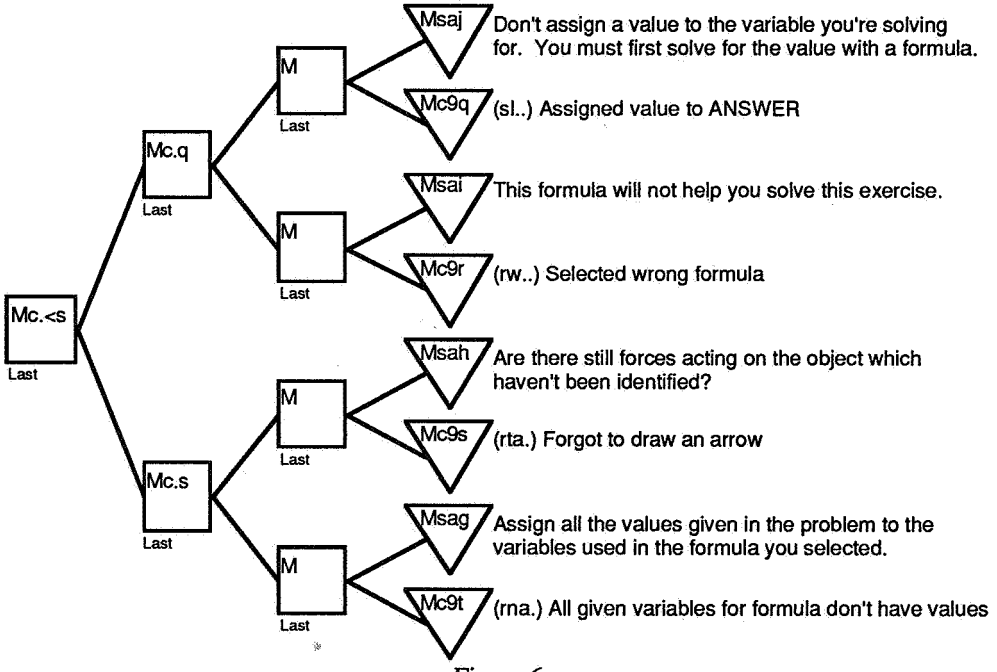


Figure 6
Remediation Pattern Matching Tree

This remediation mechanism was used with great success in the Intelligent Physics Tutor. However like OPL notation, the misconceptions must be translated by hand into the tree notation shown in Figure 6. This project will implement an editor to automate this process.

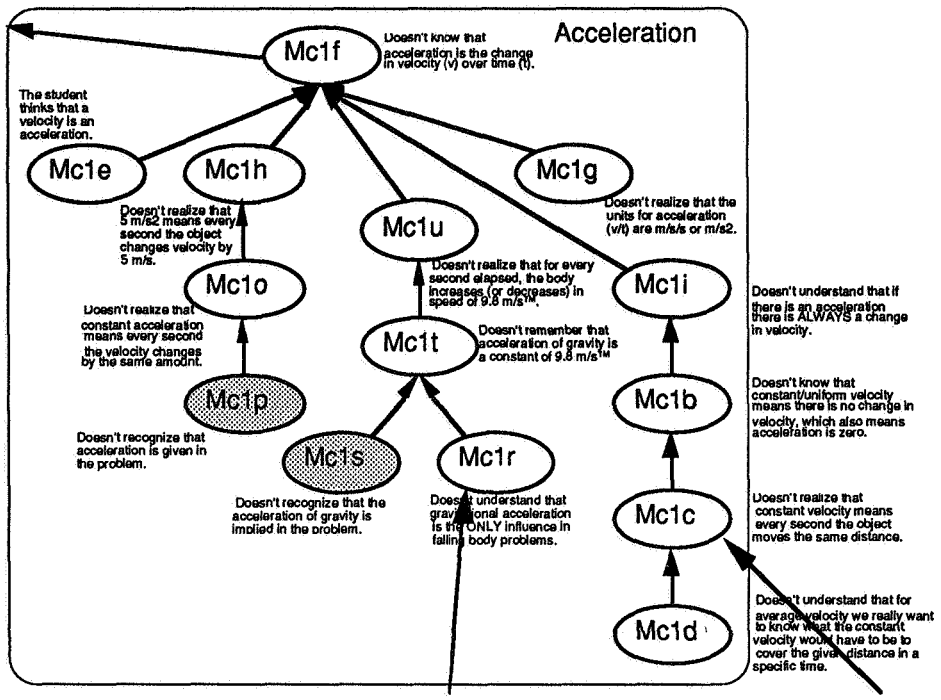


Figure 7
Related Misconceptions

Lesson Planner

The STB has developed a second algorithm, as a part of misconception theory, which allows the personal trainer to select appropriate exercises for each student. Exercise selection is based on knowledge which the personal trainer gained from watching the student complete previous exercises. This process is implemented as the Lesson Planner module.

Selecting an appropriate exercise requires indexing meta-knowledge about each exercise. Meta-knowledge includes concepts that must be understood to successfully complete the exercise and misconceptions commonly revealed during this exercise.

As the student works each exercise, the lesson planner uses this meta-knowledge to compile a list of the concepts the student has applied. The lesson planner also builds a list of misconceptions diagnosed by the remediation module. Using these two lists, the lesson planer creates a third list recommending concepts for future study. Selecting exercises simply becomes a matter of matching the concepts recommended to the exercise's meta-knowledge (see Figure 7).

This technique was used with great success in the Intelligent Physics Tutor. Like OPL and misconceptions, these meta-data indexes must be translated by hand into executable data files. This project will develop a tool for graphically editing these data files.

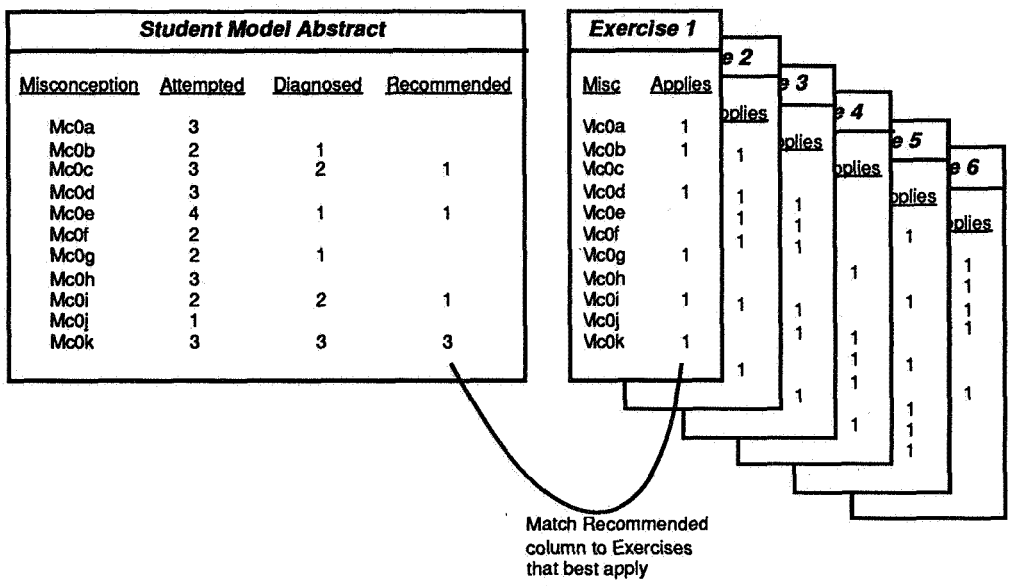


Figure 7
Meta-Knowledge Index

CONCLUSIONS

The ICAT authoring environment will allow space station trainers to develop ICAT training without outside programming help. By eliminating dedicated knowledge engineers and reusing existing simulations, we should see an inherent 50% reduction in development cost. This coupled with ICATs ability to provide more people, more training in an equivalent time, will provide a substantial cost savings in required personnel. Additionally, when completed, this environment will represent a product which could distribute NASA's ICAT technology to the private sector.

REFERENCES

Anderson, J.R., Boyle, C.F., & Yost, G. (1985). The geometry tutor. In A. Joshi (Ed.), Proceedings of the Ninth International Joint Conference on Artificial Intelligence (pp. 1-7). Los Altos, CA: Morgan Kaufmann.

Burns, H., & Parlett, J.W. (1991). The Evolution of Intelligent Tutoring Systems. *Intelligent Tutoring Systems Evolutions in Design.* Hillsdale, NJ: Lawrence Erlbaum Associates.

Hollan, J.H., Hutchins, E.L., & Weitzman, L. (1984). Steamer: An interactive inspectable simulation-based training system. *AI Magazine*, 5, 15-27.

Johnson, L., & Soloway, E. (1984). Intention-based diagnosis of programming errors. *Proceedings of American Association of Artificial Intelligence Conference* (pp. 162-168). Los Altos, CA: Morgan Kaufmann.

Richardson, J.J. (1988) Directions for Research and Applications (page 251). *Foundations of Intelligent Tutoring Systems* . Hillsdale, NJ: Lawrence Erlbaum Associates.

Riesbeck, K.C. & Schank, R.C. (1991) From Training to Teaching: Techniques for Cased-Based ITS. *Intelligent Tutoring Systems Evolutions in Design.* Hillsdale, NJ: Lawrence Erlbaum Associates.

Way, R.D. (1991) An Intelligent Tutoring System for Physics Problem Solving. *NASA's 1991 Conference on Intelligent Computer Aided Training*, NASA Conference Publication 10100, Vol.I

Woolf, B. (1991) Representing, Acquiring, and Reasoning About Tutoring Knowledge. *Intelligent Tutoring Systems Evolutions in Design.* Hillsdale, NJ: Lawrence Erlbaum Associates.

NASA Johnson Space Center Information Systems Directorate Information Technology Division

Intelligent Computer-Aided Training Authoring Environment

by

Robert D. Way
LinCom Corporation
1020 Bay Area Blvd. Suite 200
Houston, TX 77058

NASA Johnson Space Center Information Systems Directorate Information Technology Division

Road Map

What is ICAT?
How are ICATs built?
Why an authoring environment?
Authoring environment vision?

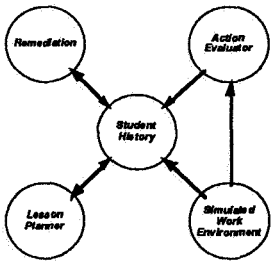
NASA Johnson Space Center Information Systems Directorate Information Technology Division

What is Intelligent Computer-Aided Training?

Simulated Work Environment
- Students learn by doing real tasks
- Presents new concepts as on-the-job training

Automated Personal Trainer
- Customizes lessons for each student
- Provides immediate feedback and help
- Summarizes student progress for the instructor

NASA Johnson Space Center Information Systems Directorate Information Technology Division



NASA Johnson Space Center Information Systems Directorate Information Technology Division

SHIFT

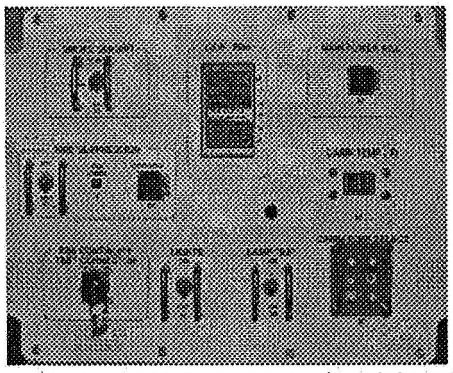
NASA Johnson Space Center Information Systems Directorate Information Technology Division

ICAT Tenets

Present the concepts in the real environment with the real objects. (Simulate the Work Environment)
Have the student do what you want him to learn.
Watch as the student does the job, take note of everything.
Watch quietly as the student works.
Give appropriate help when asked.
Point out errors while they're still in context.
Let the student recover from errors if possible.
Never give a formal "Test".

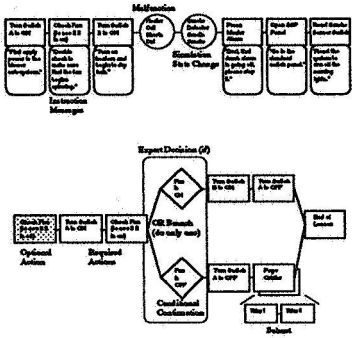
NASA Johnson Space Center Information Systems Directorate Information Technology Division

Simulated Work Environment



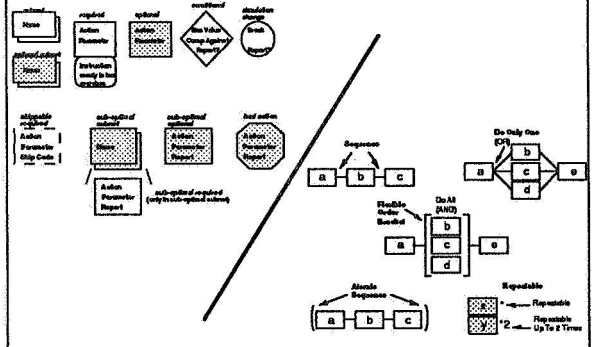
NASA Johnson Space Center Information Systems Directorate Information Technology Division

Action Evaluator Procedures



NASA Johnson Space Center Information Systems Directorate Information Technology Division

Action Evaluator Notation



NASA Johnson Space Center Information Systems Directorate Information Technology Division

Teaching Principles

Teacher's recognize a set of common misunderstandings.

They normally gear their responses more to these misunderstandings than to the students queries.

If their initial instinct appears incorrect they fall back to other commonly related misconceptions.

They double-check themselves by watching to see if the student exhibits other signs of the misconceptions.

NASA Johnson Space Center Information Systems Directorate Information Technology Division

Modelling the Teacher

Most of these misconceptions are easily recognizable from patterns of student actions.

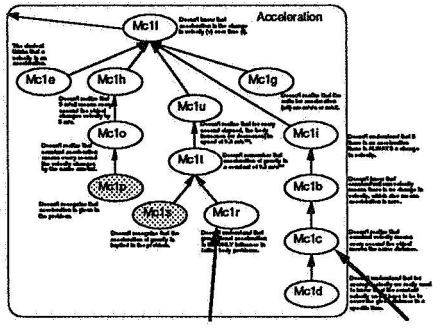
We then respond by mimicking a teacher who suspects a certain misunderstanding.

Once the misconceptions are known we are able to relate common ones as a teacher would.

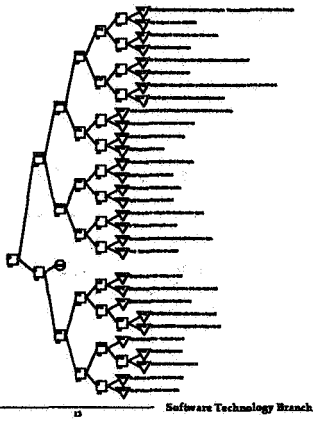
We then adjust the curriculum to double-check the student's understanding.

NASA Johnson Space Center Information Systems Directorate Information Technology Division

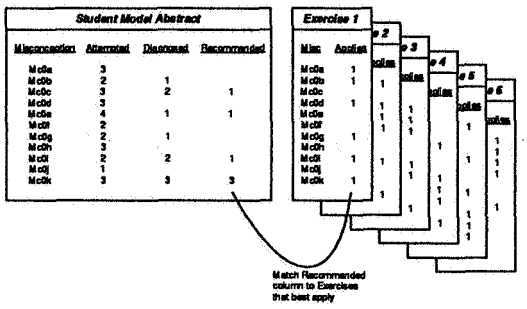
Remediation Hierarchy



Remediation Tree



Lesson Planner



Why Authoring Tools?

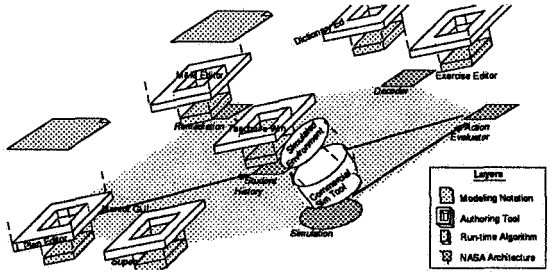
Current Systems

- Labor intensive to build
- Require programmers trained in ICAT to maintain

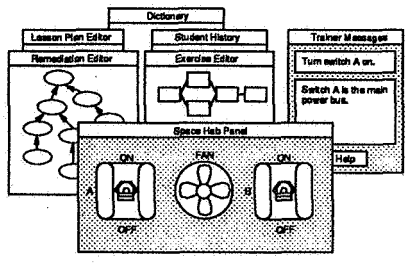
Space Station Training

- Multiple ICATs training five separate subsystems
- Must use pre-existing simulation models
- ICATs must be maintainable by without programming

Tool Architecture



ICAT Authoring Tool



Where's the Intelligence?

Immediate "teacher like" feedback to student actions.

Context sensitive help at all times.

Adjustment of material based on the demonstrated understanding of the student.

Teacher level summaries of both student and class progress.