INTELLIGENT TUTORING SYSTEMS FOR PROCEDURAL TASK TRAINING OF REMOTE PAYLOAD OPERATIONS AT NASA

James C. Ong
Stottler Henke Associates, Inc.
San Mateo, CA

Steven R. Noneman, Operations Training Group
NASA Marshall Space Flight Center
Huntsville, AL

Intelligent Tutoring Systems (ITSs) complement training simulators by providing automated instruction when it is not economical or feasible to dedicate an instructor to each student during training simulations. To lower the cost and difficulty of creating scenario-based intelligent tutoring systems for procedural task training, we developed the Task Tutor Toolkit (T^3), a generic tutoring system shell and scenario authoring tool. The Task Tutor Toolkit employs a case-based reasoning approach where the instructor creates a procedure template that specifies the range of student actions that are "correct" within each scenario. The system enables a non-programmer to specify task knowledge quickly and easily by via graphical user interface, using a "demonstrate, generalize, and annotate" paradigm, that recognizes the range of possible valid actions and infers general principles that are understood (or misunderstood) by the student when those actions are carried out. The annotated procedure template also enables the Task Tutor Toolkit to provide hints requested by the student during scenarios, such as “What do I do now?” and “Why do I do that?” At the end of each scenario, RPOT displays the principles correctly or incorrectly demonstrated by the student, along with explanations and background information. The Task Tutor Toolkit was designed to be modular and general so that it can be interfaced with a wide range of training simulators and support a variety of training domains.

We used the Task Tutor Toolkit to create the Remote Payload Operations Tutor (RPOT), a tutoring system application which lets scientists who are new to space mission operations learn to monitor and control their experiments aboard the International Space Station according to NASA payload regulations, guidelines, and procedures. NASA is currently evaluating the effectiveness of RPOT and the Task Tutor Toolkit and is exploring other potential training applications for the Task Tutor Toolkit.

James Ong is a Group Manager at Stottler Henke Associates, Inc. (SHAI) where he specializes in the design and development of intelligent tutoring systems. Over the past 19 years, James has held a variety of applied research, engineering, and engineering management positions at SHAI, AT&T Bell Laboratories, Bolt Beranek and Newman, and Belmont Research. James received an S.B. degree in electrical engineering from MIT, an M.S. degree in electrical engineering and computer science from the University of California at Berkeley, an M.S. degree in computer science specializing in artificial intelligence from Yale University, and an M.B.A. from Boston University.

Steven R. Noneman is the Assistant Lead of the Operations Training Group at NASA's Marshall Space Flight Center (MSFC). In this capacity, he is leading the payload training development in coordination with the International Space Station (ISS) payload community, the ISS international partners, mission operations controllers, and crew-members. He is a member of the ISS International Training Control Board and co-chair of the Payload Training Panel. He also leads advanced training technologies development and integration for MSFC.

Mr. Noneman has over 25 years of experience at NASA. He started working at MSFC as an Aerospace engineering co-op student from the University of Cincinnati. He later received a Master of Science degree in Systems Engineering from University of Alabama in Huntsville. During his NASA career, he worked on the Spacelab-1 and Spacelab-3 Shuttle missions and was an ASTRO-1 Payload Operations Director for the first Spacelab payload mission controlled from Huntsville. He has also led the operations development of several other MSFC projects. He is an author of several papers on payload mission operations. In 1997 he became the Chief of the Training Branch in the Mission Operations Laboratory and has served in his present position since June 1999.

INTELLIGENT TUTORING SYSTEMS FOR PROCEDURAL TASK TRAINING OF REMOTE PAYLOAD OPERATIONS AT NASA

James C. Ong
Stottler Henke Associates, Inc.
San Mateo, CA

Steven R. Noneman, Operations Training Group
NASA Marshall Space Flight Center
Huntsville, AL

INTRODUCTION

ITSs Augment Simulations with Automated Instruction

Currently, NASA employs training simulations extensively so that space and ground personnel can acquire the practice needed to achieve high levels of proficiency. Typically, these training simulations require instructors to help students operate the simulator, monitor and evaluate each student's performance, and provide help and instructional feedback to students that complements the "natural feedback" provided by the simulator. Although instructor-assisted training simulations are very effective, they require significant amounts of time from instructors to facilitate the training sessions and provide feedback to students. When students are geographically distant from the instructors, transporting students to instructors (or vice versa) consumes additional travel time and costs. When there are many more students than instructors, there simply may not be enough instructors with the appropriate expertise at any cost.

Intelligent Tutoring Systems (ITSs) encode and apply the subject matter and teaching expertise of experienced instructors to provide students with individualized instruction automatically. ITSs monitor student actions carried out within training simulations, evaluate those actions to infer missing knowledge and misconceptions, and provide specific hints and feedback to the student based upon its model of the student's knowledge. ITSs complement training simulators by providing automated instruction when it is not economical or feasible to dedicate an instructor to each student during training simulations.

Research studies carried out so far show that students taught using ITSs generally performed better and learned faster, compared to classroom-trained students. For example, at Carnegie Mellon University, researchers developed an intelligent tutoring system called the "LISP Tutor" which taught computer programming skills to college students. In one controlled experiment, students who received computer-tutoring scored 43% *higher* on the final exam than the control group who received traditional instruction. When given complex programming problems, the control group required 30% more time to solve these problems, compared to the tutored students [Anderson, 85]. The "Sherlock" intelligent tutoring system taught Air Force personnel troubleshooting procedures for problems associated with an F-15 manual avionics test station. Students taught using Sherlock performed significantly better than the control group and, after 20 hours of instruction, performed as well as experienced technicians with four years of on-the-job experience [Nichols, 92].

Authoring Tools Can Lower the Cost of ITSs for Procedural Task Training

To date, however, operational use of ITS technology has been limited, compared to other types of computer-based training. One major reason is that these systems are usually costly and difficult to develop because significant development effort is required to:

?? Encode the task knowledge needed to monitor and evaluate the student's performance, and

?? Develop the simulator with which the student interacts.

Many of the NASA training domains can be classified as **procedural tasks** for which there exist procedures, guidelines, and strategies which describe the correct set of steps to be taken within each situation. In many situations, detailed procedures exist which specify exact sequences of button presses, switch activations, and other "low-level" actions to be carried out. In other situations, guidelines and strategies specify the actions that are appropriate in various situations at a higher-level, requiring translation into the specific, lower-level actions to be carried out. Frequently, more than one procedure, guideline, or strategy may be applicable to each scenario that the student must select and apply in combination.

For example, payload operations guidelines for the International Space Station require payload users, such as scientists, to report anomalies associated with their experiments. To comply with this guideline, the payload developer must assess the situation (a payload anomaly has occurred), recognize the relevant guideline (report the anomaly), identify the appropriate procedures (use the OC voice loop to notify the Operations Controller), and recall and execute the specific steps associated with this procedure.

Training documents and classroom instruction can present these procedures, guidelines, and strategies to students and test their recall of this information. Use of these methods alone, however, can instill "inert knowledge" which students can recall but may not apply correctly when the situation clearly calls for it. In addition, as time passes, knowledge that is not used tends to be forgotten. To achieve and maintain high levels of proficiency needed to handle realistic and complex situations, students must practice the application of this knowledge in a wide range of scenarios in training situations when mistakes can be tolerated. Unlike "practice and drill" training systems which focus on the system's "buttonology" (e.g., what function does switch A perform) and the memorization of detailed procedures, scenario-based training presents hypothetical situations that require the student to assess situations; identify applicable procedures, principles, guidelines, and strategies; and determine the appropriate sequence of actions.

To reduce the cost and difficulty of creating tutoring systems for procedural task training, we developed the **Task Tutor Toolkit (T^3),** a generic tutoring system shell and authoring capability that supports economical and practical development of intelligent tutoring systems for procedural task training. The Task Tutor Toolkit was designed to simplify the development of these types of intelligent tutoring systems by:

?? Enabling instructors and subject matter experts to specify the task knowledge used by the ITS quickly and easily, without programming, and

?? Interfacing with diverse NASA training simulators to support a wide range of training domains and leverage the substantial development invested in these simulators.

Remote Payload Operations Tutor

The Telesciences Resource Kit (TReK) is a new, networking capability that will enable scientists to monitor and control their experiments aboard the International Space Station from a remote location, such as their office. Many TreK users may be new to space mission operations and relatively unfamiliar with the payload operations procedures, guidelines, and software applications.

We developed the **Remote Payload Operations Tutor (RPOT),** a specific tutoring system based on the Task Tutor Toolkit which helps TReK users learn remote payload operations guidelines in an interactive learning environment. RPOT combines simulation with automated tutoring capabilities that provide hints and instructional feedback to the student. RPOT complements existing NASA regulations and training documents by enabling payload developers to monitor and control a simple, hypothetical experiment called the Hypothetical International Space Station Experiment (HISSE) within simulated scenarios.

The Remote Payload Operations Tutor is comprised of two main software applications: the RPOT Authoring Tool and the RPOT Tutoring System. Both systems incorporate the **RPOT Simulator** that provides simplified simulations of four software applications remote payload users will use to monitor and control payloads:

?? Hypothetical TReK telemetry application for HISSE that enables payload users to monitor the values of scientific, safety, and resource consumption variables.

?? Voice over the Internet application that enables payload users to select voice loops for listening and/or talking, to coordinate their activities with those of NASA personnel. For example, students can select a voice loop, select a position to talk to, and select an utterance to say. Simulation rules can be defined to generate and responses to these utterances to emulate conversations. Utterances from simulated persons "heard" by the student are tagged with the speaker's voice loop and position, enclosed within angle brackets. (See Figure 1.)

?? Command Operations application, configured with commands that control HISSE. Using this simulated application, students can select and submit commands to control their experiment.

?? Payload Information Management System (PIMS) application which enables students to submit and retrieve reports.

The **RPOT Tutoring System** provides students with a practice-based learning environment which:

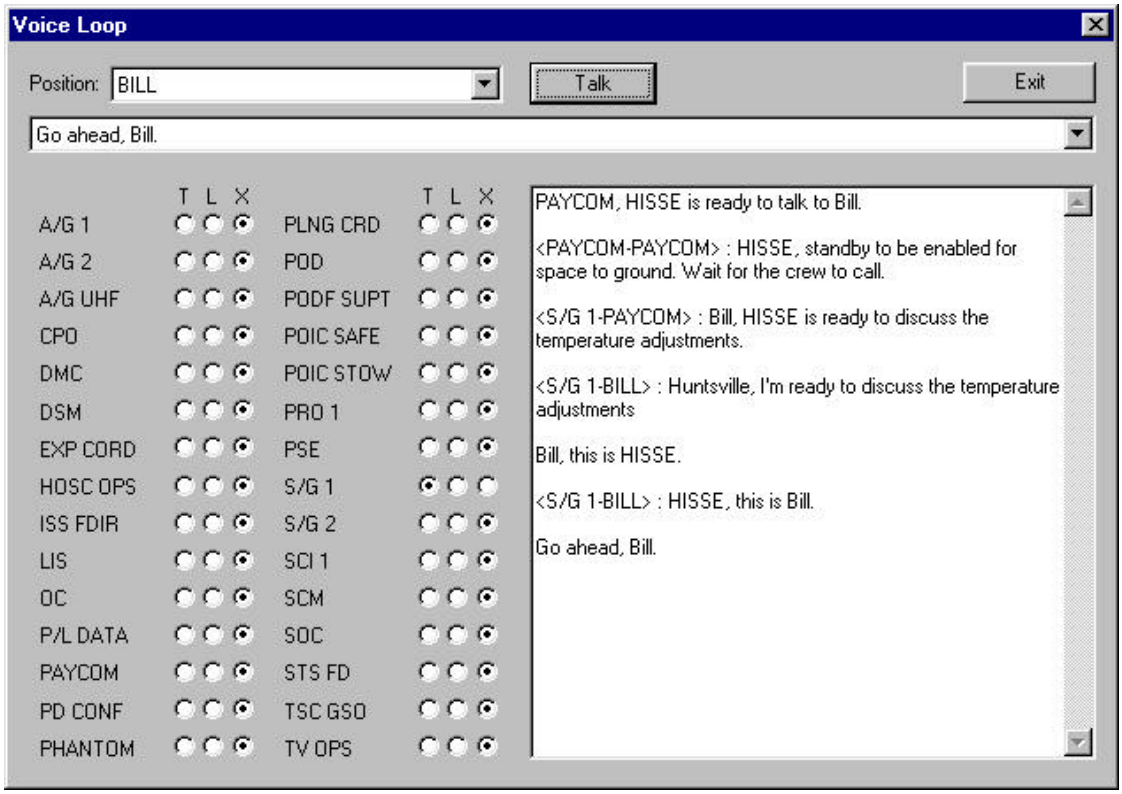

Figure 1 - Emulated Voice over the Internet Application

?? Invokes the domain-specific RPOT Simulator to present scenarios which are created by an instructor or subject matter expert,

?? Provides hints to the student,

?? Evaluates the correctness of the student's actions by comparing each action with the procedure template and evaluation rules specified by the scenario's author, and

?? Identifies and explains principles (concepts and facts) the student appears to understand or not understand.

The **RPOT Authoring Tool** enables instructors and subject matter experts (authors) to define scenarios presented to students without programming, by:

?? Using the RPOT Simulator to record a correct sequence of actions for the scenario,

?? Generalizing these actions to create procedure templates which recognize valid sequences of student actions which may differ slightly from the recorded sequence,

?? Adding evaluation rules which detect incorrect action sequences, and

?? Annotating these procedure templates and evaluation rules to infer principles that the student appears to know or not know when the student carries out correct and incorrect actions.

TECHNICAL APPROACH

Scenario-Based Instruction

The tutoring system presents scenarios that require the student assess situations, perform tasks, and respond appropriately in the context of those situations. For example, one of the RPOT scenarios presents the student with the following initial briefing:

You are the principal investigator for an experiment called HISSE (Hypothetical International Space Station Experiment). Your experiment has a low power mode and a high power mode. Usually, you run your experiment in high power mode and require a power resource envelope of 200 watts, although consumption of 150 watts in this mode is typical. Today, the crew is scheduled to run your experiment in low power mode. Your payload power resource envelope for this low power experiment is 50 watts. The crew will begin operations with your experiment immediately.

In this scenario, it turns out that the experiment was incorrectly set to run in the high power mode. The student can only detect this error if he or she has been monitoring power resource consumption using the simulated TReK telemetry application. Once the error has been detected, the student should notify the Operations Controller (OC) position using the OC voice loop. After the crew has identified the anomaly, the student should document the anomaly using the Payload Information Management System (PIMS). Thus, this scenario requires the student to apply the following payload operations principles:

?? Monitor experiment during experiment operations

?? Monitor the space-to-ground loop during experiment operations involving crew

?? Report payload resource anomalies,

?? Document payload anomalies

Case-based Authoring Simplifies the Entry of Task Knowledge

Task knowledge is used by intelligent tutoring systems to represent how a task should be carried out. Many systems encode task knowledge within a rule-based expert system which solves each problem presented to the student and then assesses each student by comparing its own reasoning with the student's reasoning. Developing an expert system is often expensive and difficult, so this approach is economically feasible only when the benefits of the tutoring system summed over all students outweigh the costs of developing the expert system.

To support cost-effective authoring of tutoring systems without requiring expert system development, WE pursued a case-based approach where a *procedure template* specifies the range of student actions which are "correct" within a given scenario. This method avoids the need to encode the ability to solve all possible problems within an expert system. Instead, a case-based approach requires only the ability to specify how the student should act within each individual scenario. Because each procedure template is specific to a single scenario, the system can employ reasoning methods that are simpler than would be required if the system had to evaluate the student's performance in arbitrary scenarios. This simplicity enables the task knowledge to be specified quickly and easily by a non-programmer via graphical user interface, rather than as a set of expert system rules developed by an AI programmer.

Modular Architecture Accommodates Diverse Simulators

The **Task Tutor Toolkit (T^3)** is composed of two T^3 software modules, the **T^3 Tutoring System** and the **T^3 Authoring Tool**. These modules interface with diverse, application-specific simulators to create specific tutoring system applications. Each application is created by interfacing an application-specific simulator with the T^3 Tutoring System (see Figure 2.) Each authoring tool application is created by interfacing an application-specific simulator with the T^3 Authoring Tool. For example, the RPOT tutoring system application is created by interfacing the RPOT Simulator with the T^3 Tutoring System. The RPOT authoring tool is created by interfacing the RPOT Simulator with the T^3 Authoring Tool. The Simulator, Simulation Interface, and T^3 modules can be incorporated within the same executable program, or the Simulator and the Simulation Interface/T^3 modules can reside in separate programs that communicate over a network.

The Task Tutor Toolkit was designed to operate with diverse set of existing and planned training simulators at NASA. To accommodate the wide range of external system interfaces supported by different simulators, the Toolkit relies on Simulation Interface software modules to mediate between the T^3 modules and each simulator.

These simulator-specific interface modules implement the inter-module or inter-process communication and message translation between the T^3 modules and each simulator to provide a standard application programming interface (API) by which the Task Tutor Toolkit can:

?? Invoke the simulator,

?? Receive notifications of user actions, and

?? Receive updated simulation state variable values.

To support diverse simulators, the Task Tutor Toolkit represents user actions carried out within each

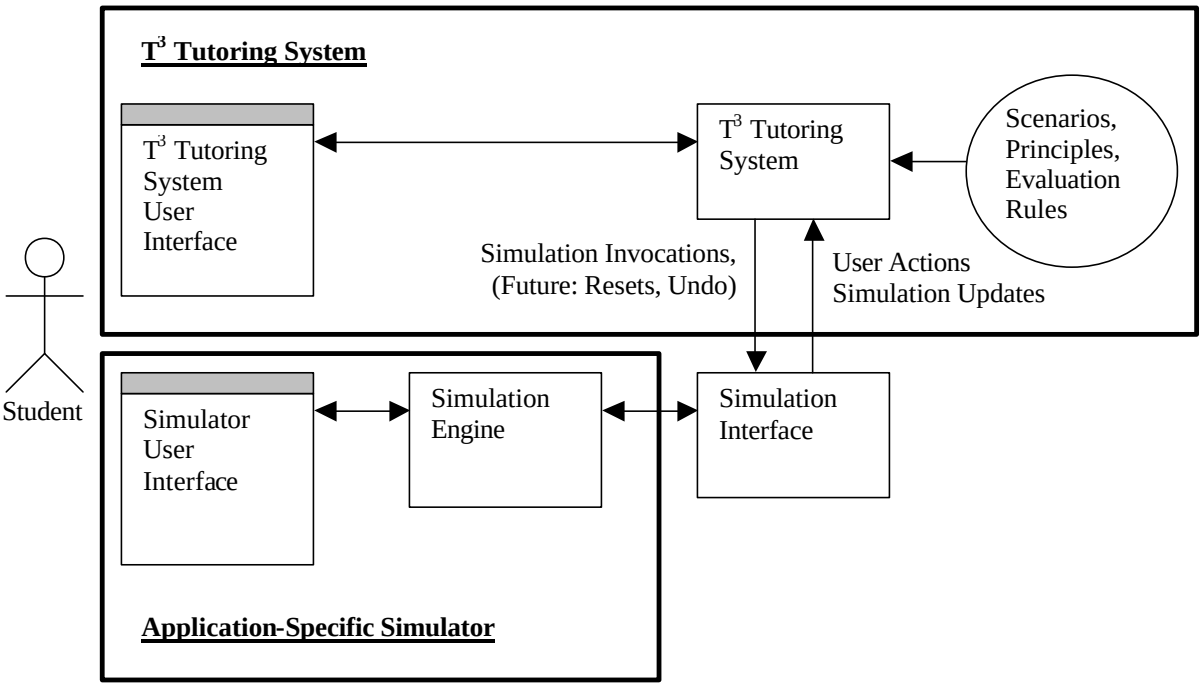

Figure 2 - Tutoring System Application =
T3 Tutoring System + Application-Specific Simulator + Simulation Interface

simulator in a general way, as tuples of string, integer, and floating point values, called *events*. For example, the action of setting the value of Dial A to 5 could be represented by the following vector of 2 strings and a floating point number, ("Set", "Dial_A", 5.0). This representation framework makes it possible for the Task Tutor Toolkit to receive and process notifications of simulation-specific user actions from the Simulation Interface, even though the Toolkit has no built-in knowledge of any specific simulator or the user actions it supports.

Procedure Templates Encode Task Knowledge

Each scenario definition contains a procedure template that specifies a sequence of correct actions, or *steps*, which specify the set of correct sequences of actions to be carried out. The T^3 Tutoring System evaluates student performance by comparing each student action with the actions specified in the procedure template. Certain actions and groups of actions in the template can be associated with principles so that the tutoring system infers that the student understands a specific principle when he or she carries out the appropriate action or group of actions.

The author creates a procedure template in five steps, by:

- ?? Demonstrating the procedure to create an initial template that recognizes the exact sequence of actions carried out by the author,
- ?? Generalizing the procedure template by relaxing ordering constraints,
- ?? Generalizing the procedure template by relaxing step constraints,
- ?? Adding simulation state constraints to steps, and
- ?? Annotating the procedure template to infer principles the student appears to know, based on his or her actions performed within the scenario.

Representing task knowledge as procedure templates is an attractive approach when it is possible to specify a procedure template that recognizes the full range of acceptable student actions. This is because the simplicity of procedure templates enables rapid and intuitive authoring by non-programmers, using a graphical user interface. In procedural task training domains where there exist well-accepted procedures, guidelines, and strategies for carrying out tasks correctly in each situation, we believe that procedure templates are sufficiently powerful for representing task knowledge.

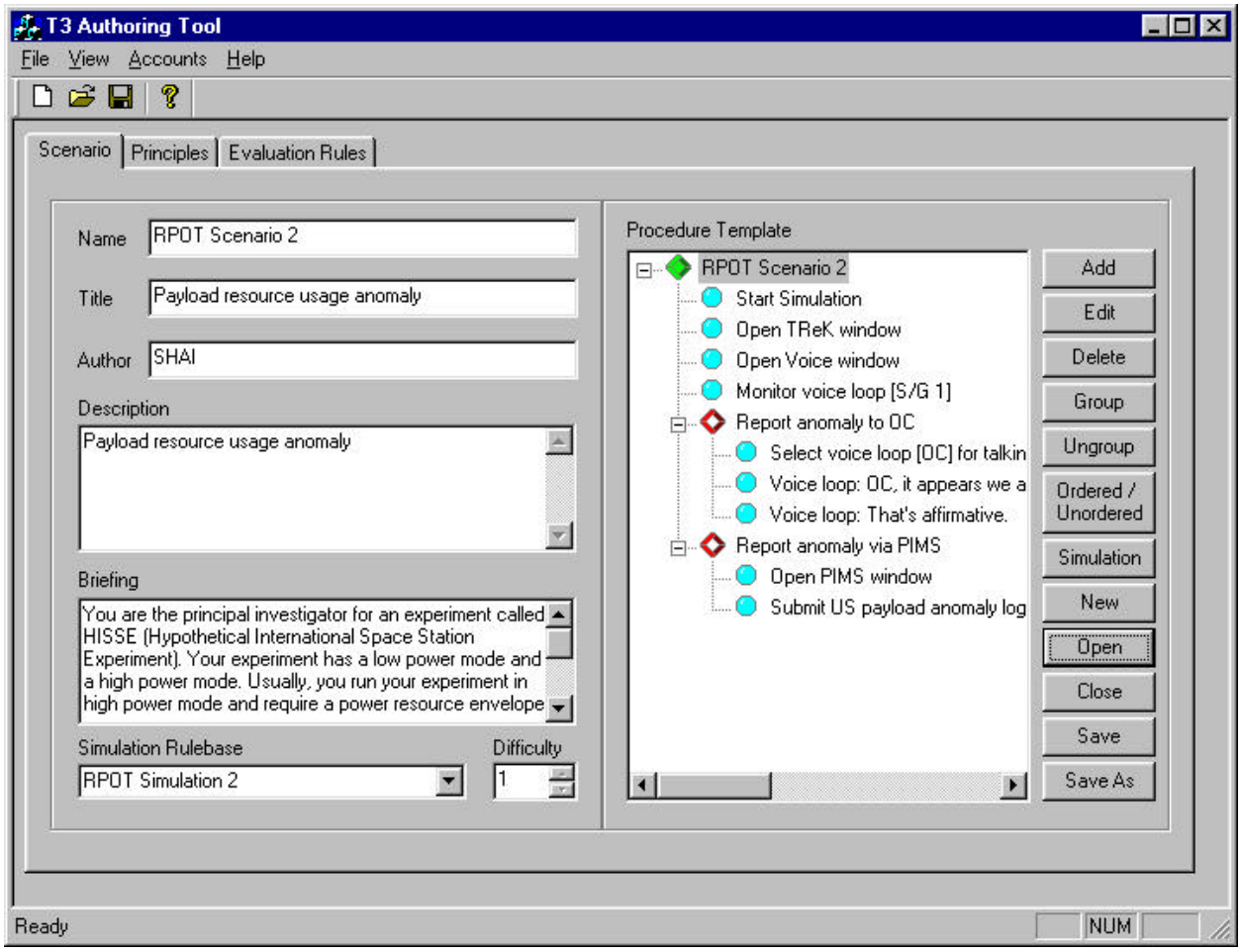

Figure 3 - T[3] Authoring Tool - Scenario tab

Authors specify scenarios using the Task Tutor Toolkit Authoring Tool (figure 3). Each circular icon represents an individual step. Each diamond-shaped icon represents a task that is comprised of one or more steps or subtasks. The instructor can select each task or step to edit its event pattern, state variable conditions, principles, and other attributes.

Demonstrating the Procedure - The instructor creates an initial procedure template by using the simulator to demonstrate a correct sequence of actions for the scenario. Tuples representing each action are forwarded to the Task Tutor Toolkit Authoring Tool to create an initial procedure template that recognizes this exact sequence of actions. For example, the instructor might carry out the following 4 actions which comprise a valid solution to the scenario:

1. Turn Power Off
2. Set Dial A to 5
3. Activate Switch B
4. Turn Power On

By demonstrating and recording this sequence of four actions, the instructor can easily create an initial procedure template that recognizes this exact set of actions carried out in this order.

Generalizing the Procedure Template - In many cases, the actions may be correctly carried out in more than one possible order. Thus, the Authoring Tool enables the instructor to relax the procedure template's ordering constraints to recognize other valid orderings. Specifically, the instructor can define tasks that contain actions or lower-level tasks (subtasks). Steps and subtasks included within a task can be ordered or unordered. Steps and subtasks within an unordered task may be carried out in any order, and steps and subtasks within an ordered task must be carried out in a specific order.

For example, if it is correct to Activate Switch B either before or after setting Dial A to 5, the procedure template can be generalized as:

1. Turn Power Off
2. Group A
   ?? Set Dial A to 5
   ?? Activate Switch B
3. Turn Power On

This generalized procedure template recognizes both of the following sequences of actions:

1. Turn Power Off
2. Set Dial A to 5
3. Activate Switch B
4. Turn Power On

1. Turn Power Off
2. Activate Switch B
3. Set Dial A to 5
4. Turn Power On

The instructor can relax constraints on individual steps to recognize other similar valid steps. For example, suppose that the instructor recorded the action of setting the value of Dial A to 5.0, and it is appropriate in the scenario to set the value of Dial A to any value between 4.9 and 5.1. The instructor could relax the *event pattern* ("Set", "Dial_A", 5.0) to ("Set", "Dial_A", 4.9 <= x <= 5.1) to accept any action that sets Dial A to a value between 4.9 and 5.1.

Adding Simulation State Condition - The Simulation Interface forwards the values of simulation state variables from the Simulator to the Task Tutor Toolkit. The instructor can specify conditions on simulation state variable values that must be satisfied in order for a step to be appropriate. For example, suppose that a step in a procedure should be carried out only after the equipment has reached a certain temperature, say 500 degrees. The instructor could associate a range of valid temperature values with the procedure step (e.g., Temperature >= 500).

Associating Principles with Step - An instructor can associate principles with tasks and steps within the procedure template. When the student carries out the task or step, the Tutoring System infers that the student has correctly applied the associated principles. For example, suppose that there is a principle that one should connect cables only when the power has been turned off. When the student connects the cable when the power is off (value of simulation variable "power" = "OFF"), the tutoring system infers that the student has correctly demonstrated knowledge of this principle.

Evaluation Rules Detect Student Errors

The author can define evaluation rules that infer principles the student has failed to apply when the student carries out actions that are incorrect under specific simulation conditions. For example, an evaluation rule might check if the student connects a cable while the power is on (e.g., value of simulation variable Power = "ON"). If this evaluation rule fires, the tutoring system can infer that the student failed the principle of turning the power off before connecting cables.

Principles Represent Student Knowledge

For each training domain, the instructor can define a set of principles which represent facts, skills, procedures, guidelines, and other knowledge. The student "passes a principle" by carrying out one or more actions which the instructor has associated with the principle within the procedure template. The student "fails a principle" by carrying out an action which is recognized by an evaluation rule associated with the principle. The student "passes a principle with help" by carrying out the correct set of actions after requesting help from the tutoring system.

The Task Tutor Toolkit represents principles simply as string values that identify each principle. Additional attributes describe the principle and identify the optional html file that contains background information about the principle.

The T^3 Tutoring System enables the student to ask three questions:

?? *What do I do?* -- The T^3 Tutoring System recommends an acceptable action (one that would be classified as Expected), and displays a short text phrase generated by the Simulation Interface that summarizes this action in the Help tab of the T^3 Tutoring System. For example:

Monitor voice loop [S/G 1]

?? *How do I do that?* -- The T^3 Tutoring System displays a short text phrase generated by the Simulation Interface that describes the specific detailed actions which the student should carry out within the simulator to carry out the recommended action. For example:

In Voice Application (Simulation Menu: Applications->Voice), select the 'L' radio button for the loop labeled 'S/G 1

?? *Why do I do that?* -- the T³ Tutoring System answers this third question by displaying the name and description of any principles that justify the recommended action, along with any additional text that may have been entered by the author for this action and scenario. For example:

> This action satisfies the following principles: Monitor the space-to-ground loop during experiment operations involving crew. (REMOTE SUPPORT SITES WITH VOICE CAPABILITY SHALL MONITOR AIR/SPACE-TO-GROUND COMMUNICATIONS DURING ACTIVE OPERATIONS OF THEIR PAYLOADS INVOLVING THE CREW. Reference: US PCC Payload Regulation U13.1-2.)

Tutoring System Evaluates Student Performance

As the student runs the simulation, the T³ Tutoring System monitors the student's progress through the scenario and evaluates each action by comparing it with the scenario's procedure template and with the evaluation rules. T³ Tutoring System classifies each action as:

?? *Expected* - the student's action matches an action described in the procedure template, the action has not yet been carried out and all actions that should precede the action have been taken.

?? *Unexpected* - the student's action does not match any action described in the procedure template, or the action has already been carried out, or not all pre-requisite steps have been carried out.

?? *Incorrect* - the student's action is recognized by an evaluation rule.

For example, consider the following procedure template:

1. Turn Power Off
2. Group A
   - ?? Set Dial A to 5
   - ?? Activate Switch B
3. Turn Power On

If the student has already carried out actions "Turn Power Off" and "Set Dial to 5", the T³ Tutoring System would classify:

?? "Activate Switch B" as Expected because pre-requisite actions "Turn Power Off" and "Set Dial A to 5" have already been carried out.

?? "Turn Power On" as Unexpected because Group A actions should precede " Turn Power On " and "Activate Switch B" contained within Group A and has not yet been carried out.

At the end of each scenario run, the T³ Tutoring System displays a Report Card to the student describing principles that were passed, passed with help, or failed, as shown in Figure 4. By pressing the Background button, the student can display a Hyper-text Markup Language (html) page or other type of document containing more detailed background information about a selected principle. The T³ Tutoring System can also display a multimedia document in html format that presents additional information about each principle. The T³ Tutoring System then updates the student model file which records the name of the scenario and a list of principles each student passed, passed with help, and failed within each scenario.

T³ Can Select Scenarios Automatically

The T³ Tutoring System enables the student to select the scenario to run from a list of scenarios. The T³ Tutoring System can also select scenarios automatically, using an algorithm that considers the scenarios already seen by the student and principles passed and failed by the student. The system favors scenarios that:

?? Have not been seen before,

?? Contain principles the student has not yet mastered, and

?? Have been classified as "easy" rather than difficult.

STATUS

We developed the Task Tutor Toolkit to support a variety of potential NASA training applications. We used the Task Tutor Toolkit to create the Remote Payload Operations Tutor (RPOT), a specific tutoring application designed to teach payload operations guidelines and procedures to scientists who are new to NASA mission operations. The software has been installed and its operation and use are being studied by the Operations Training Group at MSFC.

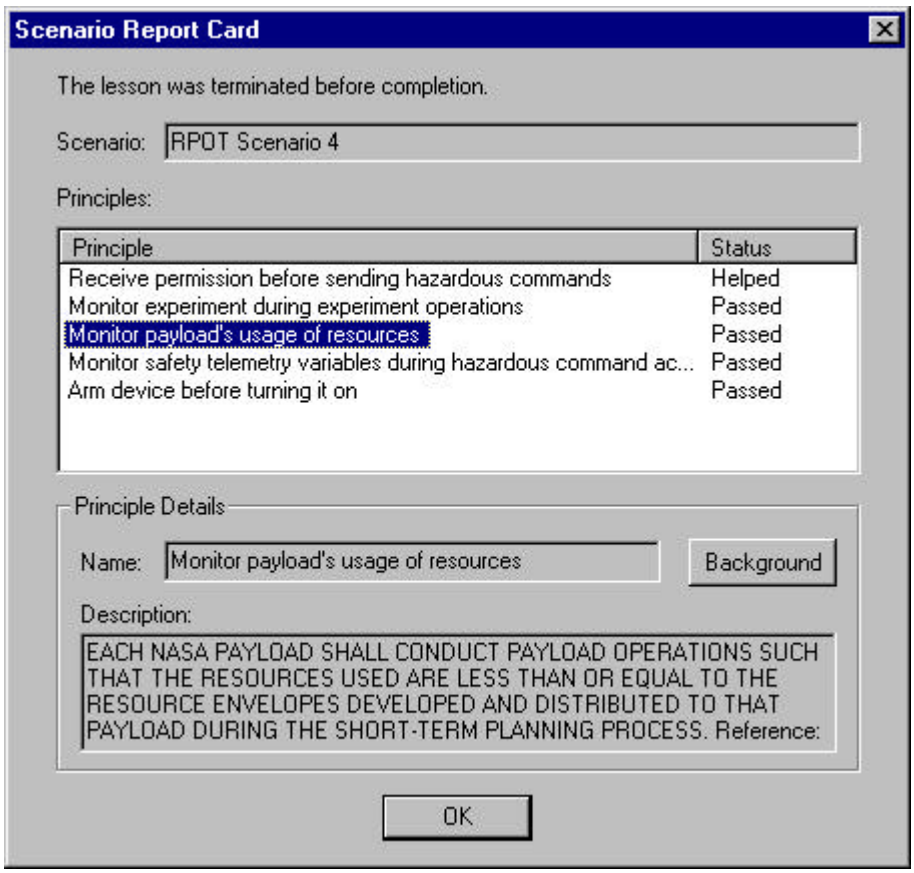

Figure 4 - Report Card displayed at the end of each scenario

NASA has prepared a preliminary evaluation plan for the system that will assess the pedagogical effectiveness from both the student and instructor perspective. A traditional training class for mission controllers covering ISS space-to-ground voice communications knowledge and skills will be the basis and reference point for the evaluation. The training scenarios used in the traditional class will be transformed using the T3. Once the RPOT with this training scenario is verified, students of varying skill levels will be asked to use the RPOT for training on space-to-ground communications and their personal evaluations of the RPOT will be solicited and reviewed. NASA also plans to evaluate the T3 utility for instructors. Factors identified for assessment are expected to include (a) the ease of capturing and revising the procedural scenarios, (b) effectiveness in introducing new knowledge and skills, (c) effectiveness in maintaining previously learned knowledge and skills, and (d) effectiveness of monitoring student progress and performance.

RPOT and the Task Tutor Toolkit runs on Intel-compatible PCs running the Windows95, Windows98, or Windows NT operating systems. By implementing an appropriate networked simulation interface, the Task Tutor Toolkit can inter-operate with single-machine or distributed simulations.

The Task Tutor Toolkit shows promise for supporting a wide variety of cost-effective intelligent tutoring systems for procedural task training. Within NASA, the Task Tutor Toolkit can be integrated with planned NASA simulators that provide system interfaces that report student actions and simulation state to external systems. In particular, the Payload Simulation Environment specifies the interface between payload simulators and instructor stations that can also be used by the Task Tutor Toolkit

Another future application is for onboard training of ISS flight crews where human instructors cannot be made easily available. Integrating an ITS with onboard computer-based training with embedded simulators may be of good training value. This same application might also serve well in the pre-flight phase when flight crews are travelling away from the high fidelity training

simulators.

NASA is also beginning to develop an Intelligence Synthesis Environment (ISE) with the goal of significantly streamlining the engineering development of space vehicles and systems. The possibilities and potential associated with inserting ITS technologies into ISE and applying them to the engineering, operations and training development environments is seen as a long range ISE area for investigation.

Outside of NASA, the Task Tutor Toolkit can be used to lower the cost of tutoring systems to teach people how to use complex hardware and software systems. In addition to learning the "buttonology" of these systems, students must also acquire the higher-level decision-making skills needed to assess each situation, identify appropriate problems and goals, select relevant equipment and software capabilities, and execute procedures to employ those capabilities.

REFERENCES

Anderson, J. R., Boyle, C., & Reiser, B. (1985). Intelligent tutoring systems. *Science, 228*, 456-462

Guralnik, D. (1996) An Authoring Tool for Procedural-Task Training -PhD Dissertation - Technical Report #71, The Institute for the Learning Sciences at Northwestern University, 1996.

Munro, A. and Q. A. Pizzini (1995) RIDES Reference Manual. Los Angeles: Behavioral Technology Laboratories, University of Southern California.

Nichols, P., Pokorny, R., Jones, G., Gott, S. P., & Alley, W. E. (1992) Evaluation of an Avionics Troubleshooting Tutoring System, Armstrong Laboratory, Human Resources Directorate, Brooks AFB, TX.