Authoring Simulation-based Intelligent Tutoring Systems

Terrance L. Goan and **Richard H. Stottler** and **Andrea L. Henke**

Stottler Henke Assoc. Inc.
2016 Belle Monti Ave
Belmont, California 94002
{goan, stottler}@shai.com

Abstract

Intelligent tutoring systems (ITSs) have shown great promise in military training domains (among others) where they can achieve many of the same benefits as one-on-one instruction, in a cost-effective manner. However, the development of real-world simulation-based ITSs is hampered by the knowledge acquisition bottleneck and difficulties creating instructive simulation scenarios. In this paper we describe an innovative approach to simulation-based ITS development using a case-based reasoning (CBR) paradigm and unique scenario generation techniques that largely circumvent the difficult and time-consuming process of knowledge elicitation. A case-based approach to knowledge gathering is highly intuitive and greatly simplifies training course authoring. Additionally, we describe how the scenarios generated with our approach can offer substantially improved realism and instructional quality through the use of intelligent simulation control techniques.

1.0 Introduction

In complex domains, instruction is often complicated by the need for the student to master a variety of concepts and to apply them in unique situations and in different sequences. In these kinds of domains, the student must develop not only a competence in the relevant facts and skills, but also an understanding of the concepts underlying these procedures. Instructional courses must be attuned to the trainee's background and needs, motivate him to develop an accurate and thorough understanding of the subject matter, and then effectively verify the correctness of his understanding and remediate inaccuracies.

When students are required to be flexible in their understanding of principles and potential applications, the most effective teaching strategy is to maximize the role of the teacher to a one-on-one interaction. In fact, [Bloom 1984] describes the two-sigma problem as the fact that students receiving one-on-one instruction perform two standard deviations better than students receiving conventional instruction. One-on-one instruction maximizes the adaptability of the instruction process, to help the student construct and test a mental model on different circumstances. The student can ask specific questions of the instructor, and the instructor can respond with immediate answers and supporting examples, tailored to the individual student. This interaction is also effective because the teacher can gauge the student's learning speed and performance, and tailor the introduction of future concepts accordingly. The instructor can also ask specific questions and present specific examples or tests to counter suspected inaccuracies in the student's mental model. Unfortunately the financial and human resources are simply not available to provide this kind of one-on-one instruction for many complex domains.

The use of an intelligent tutoring system (ITS) for individual and team training achieves many of the same benefits as one-on-one instruction. Additionally, an ITS can promote instructor productivity, cope with the higher skill requirements for trainees, provide tailored instruction and remediation, while simultaneously allowing flexibility in teaching methods. And, when ITSs are enhanced with interactive simulations, students demonstrate greater motivation during training as well as greater retention of course material. Thus, a simulation-based ITS provides instruction that is at once cost-effective and potentially superior to traditional classroom teaching methods. In this paper we describe an innovative approach to simulation-based ITS development using a case-based reasoning (CBR) paradigm and unique scenario generation techniques which largely circumvent the difficult and time-consuming process of knowledge elicitation and realistic scenario development. Additionally, we describe how the scenarios generated with our approach can offer substantially improved realism and improved instructional quality through the use of intelligent simulation control techniques.

2.0 A Case-Based Approach

ITSs have shown great promise in numerous training domains. However, the key to the effectiveness of this instruction is the technical knowledge the system contains. Traditional approaches to development of ITSs are hampered by the knowledge acquisition bottleneck—the need to construct an explicit expert mental model. We present an innovative approach to ITS development using a case-based reasoning (CBR) paradigm. A case-based approach largely circumvents the difficult and time-consuming process of constructing an explicit expert mental model. We do not need to develop an expert system that exactly models the domain expert and exhibits identical expert behavior, nor do we need to precompile a body of expert knowledge that anticipates all student interactions and errors. Instead, one or more experts' knowledge is contained in a collection of examples. The expert is asked to present a problem and its solution, with an explanation of the steps required to produce the solution. The explanation refers to principles or concepts underlying the example. Thus, the principles serve as the organizational structure of the knowledge, and the examples serve to illustrate concepts. Because a case-by-case approach to knowledge gathering is highly intuitive, authoring of the training course is greatly simplified and requires no special computer training. Further, maintenance of the ITS and the addition of updated course material is accomplished primarily through addition of new cases.

A case-based approach offers the further benefit of automatically or semi-automatically generating the student's mental model. The only completely accurate way to assess a student's mental model of a given domain is to combine performance records from specific scenarios with responses to specific cross examination, which consists of questions designed to get a direct explanation of the motivations for the student's actions and reactions in a given situation. In other approaches, this knowledge is then precompiled in an attempt to anticipate all the mistakes that a student might make. However, this kind of analysis can be extremely cumbersome, both for the student and the instructor, so it is a worthwhile goal to automate this procedure by developing an analytical technique for inferring the structure of the student's mental model from performance records. Then, active questioning of the student is used simply to verify his understanding.

There are a number of obstacles to the automation of the analysis of the student model. First of all, a student may accidentally select the correct procedures without having a thorough understanding of the domain. Likewise, a student may act incorrectly due to one of several shortcomings in his mental model of the domain. In either case, it would seem that direct questioning would be the only way to determine whether the student's performance reflects true understanding. Secondly, the student may have developed a skill with the procedure in the testing environment, but not have sufficient understanding of the overall domain to be able to apply learned concepts in new situations.

However, the intelligent use of examples is a potential way to counter these difficulties with the assessment of a student mental model. By presenting exercises requiring knowledge of principles and counter principles, it is more likely that correct student performance reflects true understanding. For example, in the domain of aircraft detection avoidance, it may be better in one situation to fly over land, and better to fly over water in another. If the student truly understands the motivations for flight path decision making, he would choose correctly in both cases. Otherwise, if he always simply chooses to fly over land, he may accidentally select the correct choice in the first case but will select the incorrect choice in the second case, thus indicating the problem with his mental model. If examples are automatically indexed by the various aspects of a domain representation that they correspond to, they can be used to identify these kinds of shortcomings in the students' understanding. Furthermore, the more examples presented to the student, the more likely it is that the student will be able to apply concepts in new situations without linking parts of his mental model to the specific contexts of training examples. If these examples are not mere text, but sophisticated multimedia scenarios, simulations, animations, 3-dimensional graphics, audio, video or hypertext, they will offer greater realism for the student, engage his interest more fully, and ultimately result in superior training and retention of concepts.

The validity of the case-based approach has both intuitive appeal and empirical backing. As early as 1940, [Gragg 1940] argued for case-based instruction. By presenting (often with a simulation) cases which illustrate the important principles, the student can see how principles are applied in operational contexts and tasks. It also defeats the well-known problem of inert knowledge first described by [Whitehead 1929] and frequently validated by other researchers. Inert knowledge is information or principles that a student knows and can recall, but which he does not apply when the situation clearly calls for it. Case-based instruction (and related concepts such as anchored instruction, scenario-based instruction, simulation-based instruction, and situated instruction) overcome this problem by showing students the application of principles in an operational setting and forcing them to apply them as well.

Case-based instruction embodies other relevant theories of instruction. For example, research has shown [Farquhar, et al. 1992] that in dynamic environments, the provision of graphical dynamic simulations improves the development of a proper mental model in students. Thus, the use of scenarios, presented graphically and dynamically is important to illustrate related principles. Another theory of instruction is situated-learning, where the importance of tying learned knowledge to tasks in an operational environment is emphasized. Again this supports the use of a simulation that approximates the operational environment for which the student is being trained.

In the next section we describe our general ITS architecture and how we implemented a simulation based ITS for Naval tactics training. We then describe our ongoing efforts to create an authoring tool that will allow Navy instructors to create and maintain their ITSs as requirements change over time.

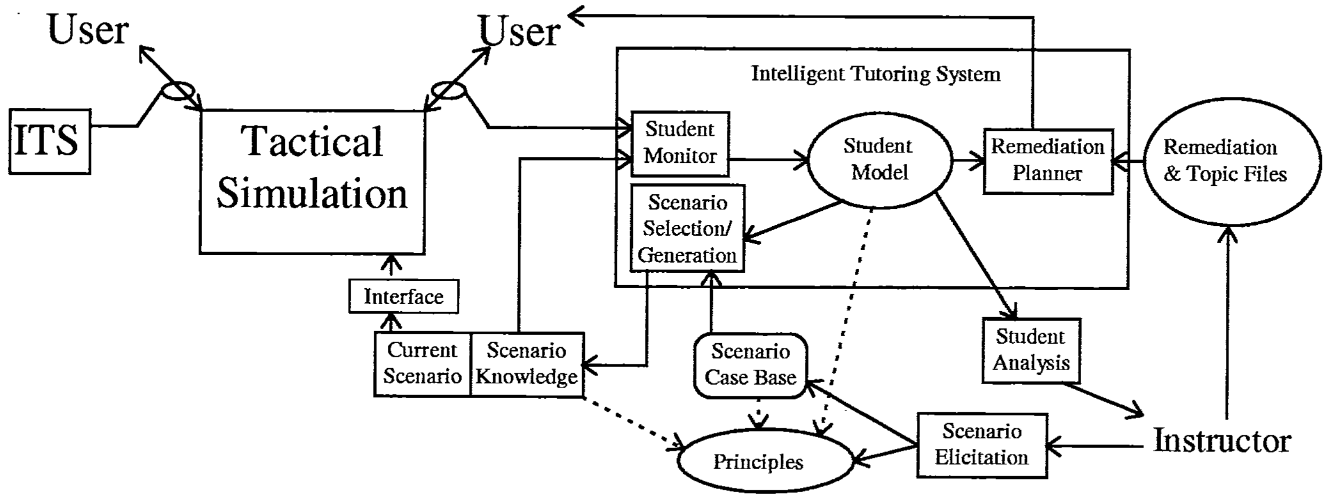

Figure 1. System Overview

3.0 A CBR-based ITS Authoring Framework and a Specific Example

The initial focus of our project was training Navy tactical action officers (TAOs). This domain is of particularly high priority to the Navy and offered a very rich task for which to explore intelligent tutoring system authoring techniques. The ITS architecture we developed is shown as a component of the overall authoring system shown in Figure 1. The student interacts with the ITS through either a simulation or through the remediation planner. The ITS Student Monitor gathers information about student actions during simulation runs and writes performance records into the Student Model. This student model will maintain information about the student's actions and decisions during different exercises. Included in this model is information about how the student performs on the principles, procedures, and techniques which have been presented in the exercises. Based on the pattern of his unsatisfactory performance on exercises as well as knowledge elicitation, the ITS will form a hypothesis as to what information the student does not understand. This hypothesis can then be used by the ITS along with information about the student's particular learning style to select remediations. After instruction, the ITS can then retrieve or generate a new scenario to retest the student. This iterative process will provide the student with a course of instruction tailored to his individual needs.

The general ITS architecture developed for this project not only resulted in an implementation in a particular domain (TAO training), but also in a reduction in development time for future ITSs in other domains. The primary components of this ITS framework are discussed below in greater detail.

3.1 The Student Model

The Student Model is perhaps the most important component in the ITS architecture. For a particular domain or course, the student model contains both basic information and derived (inferred) information. The basic information includes which scenarios the student has seen, his performance in these scenarios, as well as which remediations the student has received. Additionally, information about the success of specific remediations and remediation styles in correcting student behavior is inferred. The whole process of evaluating the performance of students is made more challenging by the fact that our ITS framework utilizes "intelligent simulation" (see below) instead of the highly scripted scenarios used in many other Navy trainers. It will be necessary to keep a log of significant simulator events which can be later scanned to determine which principles the student faced in a particular scenario run. This is because the events that occur in an intelligent simulation depend largely on the actions of the student. For example, if a student successfully evades detection, he will not be tested on principles related to defending the ship against an incoming missile.

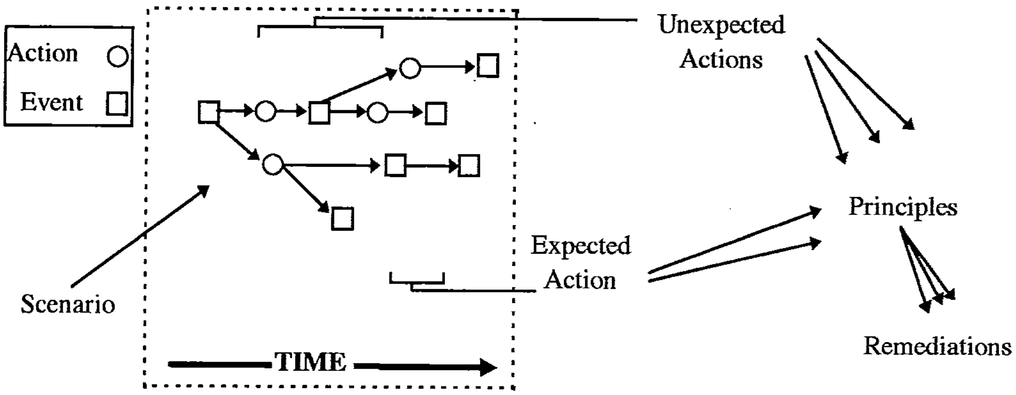

Figure 2. Connecting Student Actions to Remediation Through Events and Principles.

Within our architecture, simulation scenarios have associated "expected actions" (and "unexpected actions") in response to different types of events; expected actions are tied to principles to be taught; and these principles are tied to several remediations. An expected action is an action the instructor would hope the student would take in response to a particular situation. Unexpected actions are those actions that are deemed inappropriate for a particular situation. Both expected and unexpected actions may have an associated time period of action applicability. For example, the principle Reduce RCS (radar cross-section) was attached to the expected action of turning the ship to one of four preferred angles with respect to an incoming missile. This action was in turn tied to the event that a hostile platform has fired a missile at the student's ship. If that event occurred, then the ITS activated this expected action and principle. If the student took the correct action (within the allotted time), he received credit for this principle; but if he did not (or was too slow to protect his ship), he was considered weak in this area. These constructs consist of possible events, expected actions, and relevant principles only needed to be defined once and then could be reused across scenarios. Based on the possible events that actually did occur in a scenario, the ITS could compile the lists of principles utilized and the principles failed for the exercise and then incorporate them into the student model.

Our ITS builds mental models for students based on performance in tactical scenario exercises. The student model itself describes the student's areas of strength and weakness through the use of a "principle" hierarchy developed with the help of Surface Officer Warfare School (SWOS) instructors. This hierarchy provides an effective data structure for monitoring student learning progress as well as reasoning about the learning styles of specific students. Figure 3 presents a portion of the unclassified TAO principle hierarchy used in developing our initial prototype. This taxonomy allows the developed ITS to appropriately determine the depth of the student's knowledge (e.g., does the student understand the general principles related to weapon assignment?). Additionally, by maintaining statistics on the success rate (for individual students) of different styles of remediation associated with principles in the hierarchy, the ITS is able to tailor the instructional dialog to suit the student's learning style.

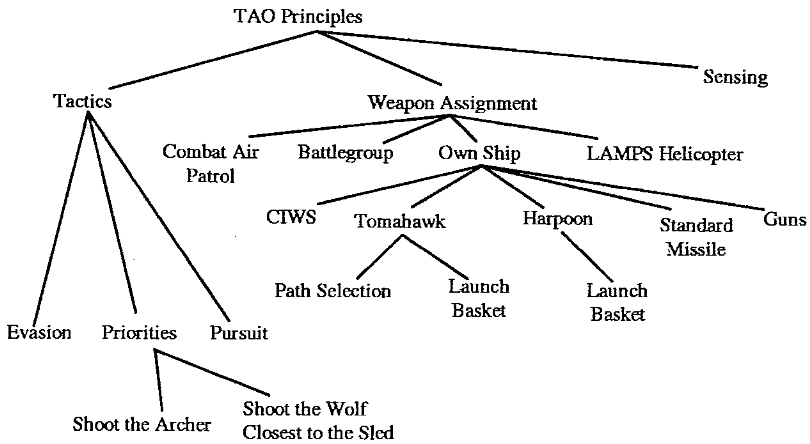

Figure 3. A Small Portion of the TAO Principle Hierarchy

In developing our TAO ITS we developed a unique approach to propagating evidence (as to the success of remediation techniques) throughout the principle hierarchy. Our approach is based on two insights. First, when selecting a remediation for a specific student on a specific principle, the system must consider: the success rate of specific remediations on that specific student (i.e., did it fail to correct the student's behavior previously?), the success rate of remediation styles on that student (e.g., does the student generally learn better with diagrams?), the success rate of specific remediations on students at large, and the success rate of remediation styles on students at large. Second, when propagating evidence through the hierarchy it is useful to recognize that similar principles are often best taught through the same medium (e.g., animation sequences and historical context to teach specific types of tactics).

In the future, we will extend our evaluation procedure by observing student actions in a series of scenarios—our ITS will determine how the student's problem solving strategies differ, if at all, across situations. This type of assessment will better allow the ITS to select appropriate remediation techniques and appropriate scenarios for testing. Additionally, techniques are required to determine the underlying reasoning used by the student in selecting his course of action. Understanding the student's underlying reasoning is key to developing corrective instruction. For this reason, we see the addition of knowledge elicitation techniques as a valuable supplement.

3.2 Scenario Selection

The Scenario Selection module selects scenarios to implement a particular training or tutoring strategy for a particular student at a particular time in his development. This module takes as input the training requirements including the current objective for the student, his current level of experience, what he has mastered and needs to master, and what knowledge the ITS still needs in order to assess the student. This information regarding the student model is used by the Scenario Selection module as a target for retrieval from the scenario case-base. Desired teaching style could also be used as input to the selection process. Teaching style includes such learning themes as whether the scenario should attempt to coax the student into an inappropriate action or attempt to divert the student from an action he should take; or whether to stress the student's cognitive load or isolate a particular principle in a simulation from as much external distraction as possible. Other influences may come from the domain expert specifying the level of expertise of each case/scenario (whether it is appropriate for novice, intermediate, or expert students) or specifying a priority ordering on principles to be taught.

Typically the Scenario Selection system would retrieve for novices, a wide breadth of scenarios to give students an appreciation for the diversity of the domain and to build up the student model as quickly as possible. This would allow the ITS to reduce the number of inappropriate scenarios that are presented to a student by quickly separating areas of expertise from areas requiring additional training. Another typical retrieval task would be to retrieve two scenarios which are both very similar to each other but different in one aspect. These pairs allow for better diagnosis of student problem solving ability by observing the difference in the student's behavior. Alternatively, it might be appropriate to retrieve a set of very similar cases, especially for students near the expert level. These show the student the nuances, the effects of small differences in a scenario and allow him to refine his skills. Finally, scenarios may be selected that return a student to subject matter previously considered mastered. This might occur if a student later unexpectantly fails a "mastered" principle (or a principle related to one thought to have been "mastered").

Scenario selection within our ITS architecture is largely based on Case-Based Reasoning (CBR) concepts, where the cases presented to the students are scenarios. The ITS automatically retrieves appropriate scenarios for the current student based on the current model of the student. The use of CBR allows the ITS to tailor itself without requiring the developer to foresee the order in which users progress through scenarios and remediations. We do not know which principles will be missed nor the order in which they will be missed, so we do not know beforehand which remediations will be used or the order in which scenarios will be presented. With CBR we do not need to consider beforehand which scenarios will be appropriate for different students in different parts of the course. The ITS decides this dynamically and automatically. Using CBR, the ITS can identify and retrieve the most appropriate one for the particular student at the particular time.

The output from the Scenario Selection/Generation module is a simulation independent scenario which is converted by the simulation interface into a scenario that can be run in a particular simulation.

3.3 Scenario Execution

Most current tactical trainers used by the Navy use highly scripted scenarios where events occur in a rigid and predictable fashion, or scenarios where simulation entities must be controlled by support staff. This approach to automated training suffers from multiple drawbacks. The most obvious drawbacks are that these types of simulations frustrate students and are not perceived as realistic and therefore reduce the teaching potential of the trainer. This frustration arises from the students' inability to alter the outcome of the scenario regardless of how well they execute their actions. A second and related problem with these simulations is that students feel they are at an unfair disadvantage because the simulation entities (e.g., opposing forces) have unrealistic access to information about the simulated world.

Our proposed approach presents a whole new way to think about tactical simulation. Within our intelligent simulation framework, entities in the simulation are viewed as intelligent agents with realistic sensor models, individual missions and behaviors, varying levels of proficiency etc. This approach makes for a more challenging and instructive simulation by adding an unprecedented level of realism. Since simulation entities only have access to the information available through their simulated sensors and take actions based only on what these sensors tell them about the student's ship's location, students will experience a higher degree of realism and a greater feeling of control.

4.0 ITS Authoring Tool

The generic ITS framework described in Section 3 provides a necessary starting point for the creation of an ITS authoring tool. The ITS authoring tool we are developing will allow domain experts to create specific ITSs without the need for programming skills. This will be accomplished through the use of a friendly user interface that guides the user through the entry of knowledge, scenarios, principles, and descriptive information. The information provided by the user will serve to instantiate a generic ITS framework to form a specific ITS that suits the instructor's requirements and will intelligently tailor instruction to student needs and learning styles.

There are several software components that will be required to elicit the ITS's domain knowledge which is stored in three locations: scenarios, the principle hierarchy/student model, and the collection of remediation files. Our overall approach to automated ITS authoring is based on a simple prompting mechanism, by which a course designer could enter new cases (see Sections 4.1 and 4.2) and the associated course information. Examples would be automatically indexed by the components of the domain that they correspond to, which will be treated as principles for training purposes in the ITS. Each new case may refer to principles that are already in the ITS, or to new principles, in which case the designer can be prompted for the information associated with these new principles (i.e., related remediation materials, and any relevant relationships with other principles (e.g., hierarchical, conflict etc.)). Finally, our authoring tool will support the creation and addition of remediation material (e.g., animations, diagrams, text messages, etc.) via a large number of existing tools.

4.1 Scenario Creation

A significant part of our effort to design a simulation-based ITS authoring tool, will be the creation of mechanisms to allow domain experts to create, edit, and test scenarios and scenario prototypes for intelligent simulations. We propose a graphical tool that will allow Navy personnel to create scenarios through an intuitive step by step process. Through the use of graphical drag and drop operations, domain experts will be able to design the initial setup of the scenario including the placement of land masses and the specification of environmental conditions. The domain expert would then specify the control mechanism for the various simulation entities. This specification process will consist of the simple connection of pre-existing "behaviors" drawn from a library. These behaviors will represent actions such as conducting an air search, or pursuing a target. The domain expert will simply describe through a graphical user interface, what actions are triggered by what events. For example, an enemy aircraft can be instructed to search a particular area in the theater of operations, and conduct a missile attack on the student's ship when (or if) the student's ship is discovered.

The other significant component of scenario creation will be the incorporation of learning objectives. Once the simulation entities have been laid out (as described above), our scenario generation tool will allow users to specify a set of expected student actions. These expected actions will be associated with simulation events (e.g., the student's ship detects an incoming missile), an instructor-specified time frame within which the student should act (e.g., 1 minute), and a set of principles associated with the expected actions (e.g., ship defense principles).

4.2 Automatic Scenario Generation

There will be times when the current case-base does not contain a scenario which adequately satisfies the selection criteria. This would tend to occur when the student has already seen several scenarios in a particular area and for testing, the ITS needs a scenario that the student has not seen before. In these situations, it will be very useful to be able to automatically generate a new scenario. This is most easily performed using scenario prototypes which only define the central learning events. A single prototype can then give rise to an almost infinite number of individual scenarios by simply filling in additional detail (e.g., geography, ambient ship traffic etc.), though they will all tend to exercise the same principles and contain the same themes. After a scenario is generated, it becomes part of the growing case base for future reuse.

A special case of scenario prototypes are scenario fragments. A scenario fragment constitutes a small fraction of what would normally be found in a scenario. Such fragments would generally be well focused on teaching a particular principle or group of principles. For example, a scenario fragment may call for an incoming hostile aircraft which launches a missile. This particular fragment would specifically test the student's ability to protect his ship. Fragments can be individually retrieved and utilized. Complete scenarios can also be created from fragments or from combinations of fragments and existing scenarios. These scenario fragments would not be created by the instructor, but rather they would be identified and extracted from existing scenarios. Since both scenarios and scenario prototypes, when actually instantiated and used, will tend to cause certain events to take place and corresponding principles to be exercised, and because the ITS keeps the chain of events that occur during a particular execution, the Scenario Generation module will be able to extract fragments (i.e., required elements from several