Techniques for Automatic AAR for Tactical Simulation Training

Richard Stottler
Stottler Henke Associates, Inc.
San Mateo, CA
Stottler@StottlerHenke.com

ABSTRACT

One of the limitations which prevents more effective use of tactical training simulations is the need for instructors or observer/controllers to observe the student's actions in the simulation and then debrief him. Many tactical training situations require this After Action Review (AAR) to ensure that the student actually learns from the experience. Thus an instructor is required to observe each simulated scenario, limiting the number of such scenarios that can be played. An automatic AAR capability therefore greatly increases the number of scenarios each student can perform. But the development of this capability is very challenging, principally because in a free play simulation, there is an infinite number of possible outcomes, at least at the most detailed level.

Two techniques have been found to be very useful in the development of automatic AAR capabilities for tactical simulations. The most common type of tactical decisions are made in real-time during the execution in a simulated scenario. General and scenario-specific Behavior Transition Networks (BTNs) have been shown to be highly adept at determining the correctness of student decisions in real time free-play simulations in a variety of domains including Navy, Army, Air Force, and Marine Corps training applications.

Another type of tactical decision occurs during the tactical planning process that results from the receipt of an order to perform an upcoming mission. The result of this process is a tactical plan which is intended to be executed to meet stated objectives. It has been shown that evaluating the correctness of the various aspects of the plan can be accomplished through the comparison to previously entered and annotated plans specific to particular scenarios. These plans are typically both good and common bad plans.

This paper describes both techniques in detail as well as their application to Navy, Army, and Air Force tactical decision-making domains.

ABOUT THE AUTHOR

Richard Stottler co-founded Stottler Henke Associates, Inc., an artificial intelligence consulting firm in San Mateo, California in 1988 and has been the president of the company since then. He has been principal investigator on a large number of tactical decision-making intelligent tutoring system projects conducted by Stottler Henke, including projects for the Navy, Army, Air Force, and Marine Corps. Currently he is working on a Combined Arms ITS as part of the CAST Upgrade project and on an intelligent tutoring prototype for the future combat system control vehicle, funded by the US Army STRICOM. He has a Masters degree in computer science from Stanford University.

Techniques for Automatic AAR for Tactical Simulation Training

Richard Stottler
Stottler Henke Associates, Inc.
San Mateo, CA
stottler@StottlerHenkeAssociates,Inc.

PROBLEM DESCRIPTION: THE NEED FOR AN AUTOMATIC AAR CAPABILITY

There is universal agreement among tactical experts and instructors that the most important single factor for training high-quality tactical decision-making is the amount of tactical decision making practice that trainees receive. This is acknowledged throughout the military services by the importance and funding placed on realistic tactical scenario practice in constructive, virtual, and live simulations. It is also indicated by the very high value the services place on individuals with successful actual combat experience. Tactical decision-making is among the highest value and criticality of all types of decisions with dozens or hundreds of lives often hanging in the balance along with often millions and sometimes hundreds of millions of dollars in potential damage.

Throughout the services most instructors agree that their trainees are not as good at tactical decision-making as they would like and that substantially more tactical decision practice is needed. The cost of live scenario practice is prohibitively expensive to increase. This leads to the conclusion that more simulated tactical scenario practice must occur. Unfortunately, for most tactical decision-making practice to be beneficial requires that the trainee receive some kind of feedback or debriefing as to whether his decisions and actions were correct, incorrect, and why. For example, just because the trainee's forces destroyed a particular enemy unit doesn't mean that his decisions were correct. Did he violate rules of engagement? Did he follow commander's intent (especially in the face of unexpected circumstances)? Did he take unnecessary risks or make other bad decisions? Traditionally human instructors are used to observe and debrief a trainee's simulated scenario performance and to answer these questions for him. The expense and availability of these instructors artificially limits the amount of practice that trainees can receive. Consider especially that trainees generally far outnumber their instructors.

One solution has been to use highly scripted scenarios, where a trainee's actions can be compared to a set script of expected actions. Unfortunately trainees often find scripted scenarios to be unsatisfying and unrealistic and therefore disregard the outcomes and feedbacks. For practice to be beneficial requires that the simulation be realistic and therefore free-play oriented. Otherwise students get frustrated that their actions don't seem to matter. Similar problems result if the students perceive the opponents as being omniscient, which can occur when following a scripted scenario where, for example, a scripted OPFOR attack is unrealistic from the perspective of the knowledge available to the attacking force.

So the requirement for increased tactical decision-making practice reduces to the problem of automatically evaluating tactical decisions in free-play tactical simulations. Unfortunately there has been an assumption throughout the services that this was not possible. This has partly been due to traditional Intelligent Tutoring System design which assumed that to evaluate a student's decisions required the construction of an expert model (really an expert system) that could make the required decisions as an expert would in order to have something to compare the student's decisions to. This begs the question of why, if you have built an expert system to do something, would you still need to train humans to do it? More importantly, for most tactical decision-making domains, development of expert systems is impractical for a variety of reasons.

However the fact that automatic systems can't be developed to MAKE the correct decisions across the full spectrum of possible situations that might occur during actual combat doesn't mean that automatic systems can't EVALUATE a trainee's decisions in specific, PREDEFINED scenarios of the instructor's choosing. It is easier to evaluate than produce a decision, especially if the system can assume that the trainee has received some training so that a) he won't make a totally random decision and that b) the system is not required to provide feedback on every possible type of mistake, especially those related to commonsense and lack of even the most basic domain knowledge. Relating to a) the assumption is normally that the student's possible actions are bounded from below and above; he is neither a complete neophyte nor a super expert who might arrive at a great, novel solution that even the instructor wouldn't have anticipated. We have found that these assumptions strongly hold in a variety of tactical decision-making training domains and applications. The rest of this paper described how these assumptions are used to apply two types of automatic evaluation techniques that tend to be applicable to different phases of military operations. First, Behavior Transition Networks (BTNs) and their applicability to tactical mission execution decision-making will be described along with examples from actual automatic evaluation systems. Then Case-Based Reasoning (CBR) and its use to evaluate a trainee's tactical plan by comparing it to previous stored and annotated plans will be discussed.

REAL-TIME TACTICAL DECISION EVALUATION: BTNS DESCRIBED

Describing BTNs is best performed by first describing Finite State Machines (FSMs). FSMs have been used for various purposes in computer science since computers were first developed and are very simple in concept. An FSM is simply a network of states with specific transitions between particular pairs of states, where each transition has a from-state and a to-state. A FSM is in one of its states, the current state, at a time. Associated with each state may be software that executes while the FSM is in that state. Associated with each transition is a condition. If that condition is true when the FSM is in the from-state of the transition, then the FSM will transition to the to-state. Generally a FSM will have one initial current state that it starts in when it first becomes active. Consider the example FSM below in figure 1 where states are rectangles and transitions are arrows labeled with ovals that represent the transition's conditions. If the current state of the FSM is "Start" and "Missile Fired at Ownship" becomes true, then the FSM's current state will become "Under Missile Attack".

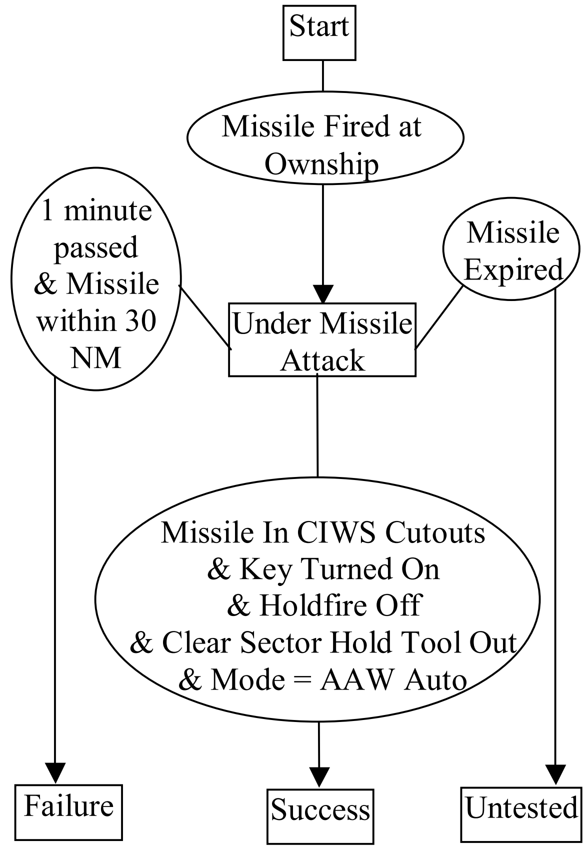

Figure 1. Example FSM (and Example TAO BTN)

FSMs are useful because the transition conditions can reference simulations events and values and trainee actions. Typically, for automatic training evaluation, half of the FSM is used to monitor events and values in the simulation, looking for a specific type of situation. This type of situation occurring places the FSM in a specific state. Then the second half of the FSM monitors and evaluates the student's relevant reactions (or lack of them) to this type of situation. Typically it writes messages to the trainee to a log file that will be presented as the AAR that describes why the actions were correct or incorrect.

For purposes of tactical decision evaluation in realistic free-play simulations, traditional FSMs have been found to be too restrictive and they have therefore been generalized into Behavior Transition Networks (BTNs). BTNs are very similar to FSMs in the sense of having states, transitions, transition conditions, and a current state, but BTNs have additional capabilities. For example, software code can be associated with the transition and execute when the transition occurs. This is useful to "spawn" additional copies of the BTN. For example, in the figure above, the transition from "Start" includes the "spawn" option so that when a missile is fired at ownship and this BTN transitions to the "Under Missile Attack" state another copy of the BTN is created, placed in the "Start" state and begins watching for the next (possibly concurrent) missile attack. BTNs also have variables that are automatically bound to the events and other conditions in the transition. These variables are easily passed between states and transitions and even across BTNs. In the example above, a variable is used which is bound to the missile that was fired that caused the transition, so that the other conditions in the BTN ("Missile Expired" and "Missile within 30 NM") can unambiguously refer to the same missile.

The best way to employ BTNs to monitor real-time tactical decision-making during mission execution is to have a large number operating in parallel where each looks at the situation and student's actions from the perspective of how they handle specific types of situations or apply specific types of principles. For example, the example above is actually one of about 50 that operate in parallel in the Tactical Action Officer (TAO) ITS [Stottler & Vinkavich 2000].

The generality of BTNs for student decision evaluation spans a continuum. In the extreme a BTN may be designed to run in almost every scenario where the situation it is looking for might come up. The BTN above is one such example. In the other extreme, a BTN may be designed for exactly one scenario. BTNs will also be designed between these two extremes. They may, for example, be designed for a family of similar scenarios, scenarios located in the same geographic area or country, scenarios with certain kinds of commander's instructions, or some other identifiable commonality. Generally it is easier and quicker to develop specific BTNs for specific scenarios, but of course this effort has to be repeated often. The trade-off between spending more time on one BTN so that it can run in many scenarios or to spend less time but have it run in fewer has to be made by the instructor or whoever is developing the automatic after action review capability and will depend very much on the specifics of what the BTN is supposed to be evaluating and the logical complexity of that evaluation.

EXAMPLES OF THE USE OF BTNS IN REAL TACTICAL TRAINING APPLICATIONS

Shown below in Figure 2 is the actual screen capture of the instructor editor used to edit automatic evaluation BTNs in the TAO ITS (Tactical Action Officer Intelligent Tutoring System). Some of the labels were a little cryptic for the purposes of this paper (where the reader cannot double click on them to get more information), so this BTN was redrawn with more informative labels and shown in Figure 1 which will be the diagram that the discussion below refers to.

Following the philosophy proposed above, this BTN evaluates and debriefs one aspect of a student's tactical decision-making - does he know when and how to employ the close-in weapon system (CIWS) on his ship to defend against missile attack? This BTN initially is in the "Start" state. This state has one transition with one condition, an event that a missile is fired at ownship where "missile" is a variable which will take on the value of any missile object that will make the condition true, in this case a missile fired at ownship. Since "Start" has only one transition, if a missile is never fired at ownship, the BTN will stay in the "Start" state forever (or, more correctly, until the simulation is over when all BTNs are terminated). If no missile is ever fired at ownship, the "Under Missile Attack" situation never occurs so there is no reason to expect related actions to occur.

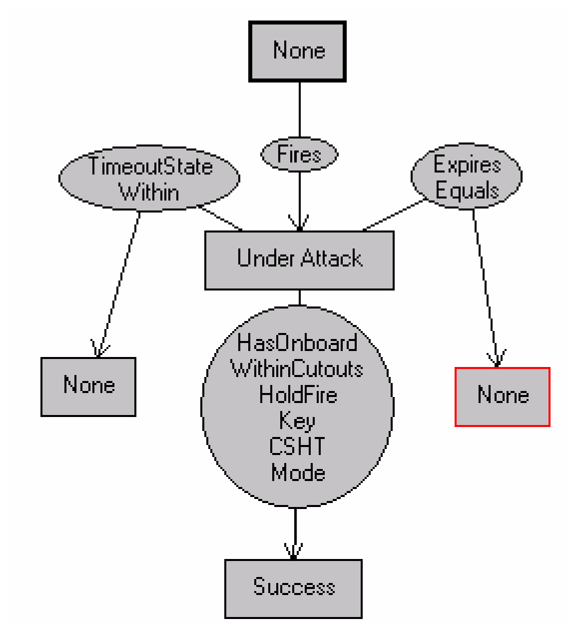

Figure 2. Actual CIWS BTN from TAO ITS

However, if a missile is fired at ownship, the "Missile" variable will be bound to it, this BTN will transition to the "Under Missile Attack" State and, since as described earlier the "spawn" option was selected, another copy of the BTN will be created, placed in the "Start" state, and will look for another "Missile Fired at Ownship" event.

Meanwhile, the first BTN is still in the "Under Missile Attack" state. There are three outgoing transitions that correspond to the three possibilities for student behavior evaluation. A successful student reaction to this situation, from the perspective of CIWS employment begins when he selects the proper CIWS (most ships have more than one with different areas of coverage, called "Cutouts"). Once he has selected the proper one, he has to make sure it is configured properly - that its key is turned on, that the hold fire is off that the clear sector hold tool is out and that the mode is set to AAW Auto. If he does all these correctly, the student has demonstrated that he understands when and how to employ the CIWS principle and the BTN transitions to the "Success" state. In addition to the condition, this transition also includes a statement (not shown) that writes a message out to the debriefing log that the student correct turned on the correct CIWS system in response to this missile attack.

Another possibility while in the "Under Missile Attack" state is that before the student correctly activated the correct CIWS, the missile got within 30 nautical miles and a minute had passed since the missile was launched, in which case the BTN transitions to the "Failure" state. In this case, the instructors considered him to have failed so that they specified that this transition should write a message to the debriefing log that the student failed to activate the correct CIWS in a timely manner. If they had desired, another transition from the "Failure" state could have checked to see if the student belatedly activated the correct CIWS. Also, if desired, other conditions could have checked for partially correct activation of a CIWS.

The third possibility is that before either the success or failure conditions are met the missile expires. This might occur, for example, if other actions by the student, such as launching a standard missile at the incoming missile, destroy it quickly. The TAO ITS included similar separate BTNs to evaluate the student's use of the Standard Missile to destroy incoming missiles, actions to reduce the ship's radar cross section, and other self-defense decisions.

A simplified BTN from an automatic AAR system to debrief F/A-18 pilots is shown below in Figure 3. This is an example of an evaluation BTN that is applicable to a specific set of scenarios instead of being generally applicable as in the last example. This BTN is applicable to scenarios where the pilot's strike mission includes at least one waypoint. This is one of about 25 that are typically, simultaneously active. This BTN starts in the "Start" state. If the pilot flies directly to the target area, he will receive the debriefing that he incorrectly bypassed the waypoint. If instead he arrives at the first waypoint, the BTN will transition to the "Reached First Waypoint" state. If he reaches the target area before configuring his radar correctly for the strike mission, he will be told of that mistake. If instead, he toggles his radar to the correct mode the BTN will transition to the "Radar Mode Activated" state. If he then reaches the target area he will receive the debriefing that he correctly performed these required mission steps. It is interesting to note that additional links are not required to handle the cases of the pilot having to respond to an air-to-air attack or otherwise being diverted from this mission. If that were to occur in the free-play simulation, then a human instructor would neither positively nor negatively evaluate the actions that this BTN is monitoring since they have become irrelevant and will not occur. And this is exactly what this BTN will do. If at any time, this mission is aborted (perhaps while the pilot engages in defensive air-to-air combat) the BTN will just be left in whatever state it happened to be in, since none of its transition's conditions will ever be true. It is also interesting to note that this BTN works for strike missions with multiple waypoints, though it would not evaluate the pilot reaching any beyond the first. Another, separate BTN could concern itself with the second waypoint and another, the third, and so on.

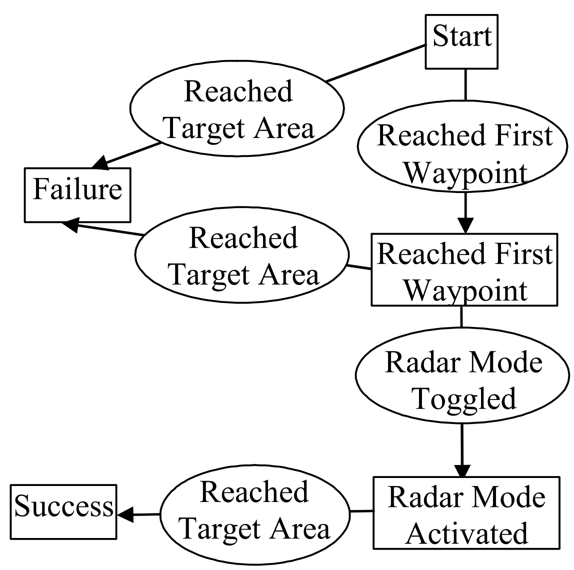

Figure 3. Sample F/A-18 BTN

A simplified BTN from an automatic AAR system to debrief Army company commanders is shown below in

Figure 4 [Stottler & Pike 2002]. This is an example of an evaluation BTN that is applicable to a very specific scenario and its specific terrain. The trainee's mission involves movement to contact (a clear indication that a bounding overwatch is called for) along a particular corridor starting from a particular ridge. A tactical expert had already analyzed the terrain and determined that there was not line of sight beyond a second ridge from the first ridge. The particular terrain features of the first and second ridge are both referenced by this BTN, which makes it specific to this particular scenario. Alternatively, automatic line of sight calculations could be performed but these are both computationally intensive and lack robustness when other factors come into play.

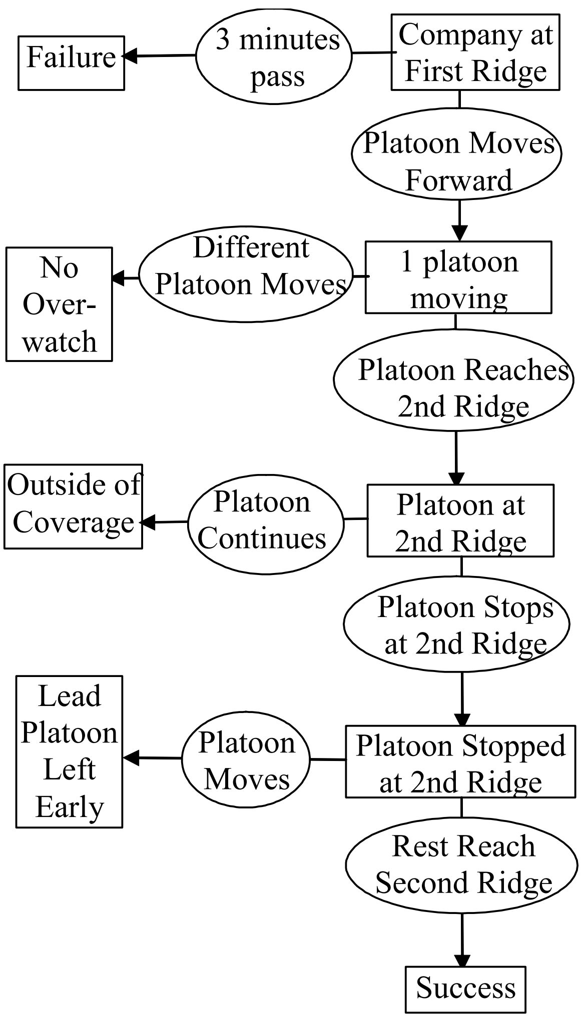

Figure 4, An Evaluation BTN for Bounding Overwatch

The initial state for this BTN is "Company at First Ridge". If any platoon fails to move forward, the student will receive the feedback that he is failing the importance of speed principle. Alternatively whichever platoon moves significantly forward first will be considered the lead platoon and the BTN will transition to the "1 Platoon Moving" state. If a second platoon moves forward before the first platoon reaches that second ridge, the student fails the bounding overwatch procedure and is told that there is not the proper overwatching element (which should consist of both the remaining platoons). Otherwise, the first platoon reaches the second ridge before any other platoon moves and the BTN transitions to the "Platoon at 2nd Ridge" state. If that lead platoon continues forward, he will be moving beyond the line of sight of his overwatching element. This causes the BTN to transition to the failure state, "Outside of Coverage" and the student is told that his lead platoon is endangering itself. Otherwise, the lead platoon stops on the second ridge and the BTN transitions to the "Platoon Stopped at 2nd Ridge" state. There are two possible transitions. If the next event is that the rest of the company reaches the second ridge, then the BTN will transition to the "Success" state, since the student both knew to and correctly applied the bounding overwatch procedure. The second possibility is that the lead platoon moves forward early, before the rest of the company arrives, in which case the student is informed of his error.

We have also developed BTNs to provide automatic after action reviews for battalion commanders in training during battalion level simulated scenarios. [Stottler et al. 2002] These BTNs evaluate tactical decisions such as when and where to send the student's individual companies and which units to assign which types of tasks during the course of the simulated battle. Some of these were scenario specific and some were more general (e.g. the commander should always make sure that a maneuver unit is supported by a mortar unit).

TACTICAL PLANS EVALUATION: USE OF ANNOTATED PLAN COMPARISON

For many tactical decision-making positions, mission execution is preceded by a planning phase. Although BTNs have been utilized to evaluate plans, this can be awkward. Another possibility is that for each scenario the instructor creates a few likely plans. This works surprisingly well for a number of reasons. The first relates to the process of tactical planning itself which involves two main stages. The first is choosing the concept of operations, the general high-level concept for the plan. The second involves filling in the details for the chosen concept. Given a specific tactical situation for which a plan must be developed, there may be theoretically an infinite number of actions a person might decide to take, however, in practice, there are only a few concepts of operation that a warfighter with even a little domain knowledge will likely consider. That is, there are only a few reasonable ways to approach the problem. Furthermore, when automatically comparing a student's plan to a plan previously created by an instructor, based on the same basic concept of operations, even if the details don't precisely match, they will almost always be similar enough for software to determine that they are based on the same concept. There may be minor differences in the size or type of unit assigned an element of a plan and possible minor location differences, but these can be handled by the techniques described below.

Another reason that having a few stored plans works is that instructors tend to design scenarios for instructional value. This usually means that performing it will demonstrate various specific principles (learning objectives). There generally are one or two correct concepts of operations that should be employed and one to three likely incorrect ways that the student will approach the problem. These are usually "traps" the students can fall into if they don't consider specific tactical principles. The result is that a handful of concepts of operations will cover all but a tiny fraction of the student plans. For example, during one particular training session, six groups of five trainees each, when presented with a particular tactical situation very similar to one described in an example below, all came up with the identical, incorrect concept of operations. Generally instructors with experience with students with a scenario can describe the likely student plans for that scenario and even the rough fraction of students likely to respond with each plan.

The comparison technique is based on having an instructor create likely annotated plans for each scenario. For a specific scenario there is generally one or two best concepts of operations to solve the tactical problem. There may be one or two other acceptable solutions and there are generally one or two unacceptable but likely student solutions. For good examples of this phenomenon see the tactical decision game solutions described in [Schmidtt 1994]. Based on each likely concept of operations, the instructor details the plan. Tactical plans generally consist of tasks assigned to different units. Generally there are locations and time periods associated with these tasks. Usually movement (changes in location over changes in time) will be involved in some of the tasks. Most military plans are described by symbols on a map. The symbol's location implies that the unit performs the task or starts at that location. Most common are unit symbols, movement arrows, and symbols defining a certain type of task (such as support by fire (SBF), attack, defend, combat air patrol, rendezvous, refuel, etc.). Detailing the plan reduces primarily to placing symbols on the 2-D map.

After the instructor has entered the plan symbols onto the map, he annotates the plan at two levels. These annotations will form the set of descriptions from which the automatic evaluation will assemble a plan debriefing for the student. At the highest level, the annotations for a correct plan describe the concept of operations and the rationale for choosing it. The descriptions and rationale are generally a combination of text and graphics. The use of html files and pointers is a convenient structure for this mixed media. Additionally the rationale may include lists of principles that the student has demonstrated he understands if he chooses this plan concept. If this plan represents a partially correct or incorrect concept of operations, both the positive (if any) and negative aspects of the plan are described in the rationale, i.e. why this concept should not have been chosen. Generally there should also be textual and machine readable pointers to the correct plan (or the closest correct plan if there is more than one). The rationale may list principles that the student has failed to apply by choosing this concept of operations.

The annotations at the lower level involve similar annotations but for each individual symbol. For each correct plan aspect (correct symbol), separate annotations describe at a minimum why the task is important and why the unit and location were selected. Again the use of html allows straightforward integration of textual and graphic rationale. Principles that the student understands if he placed the correct symbol are generally included with the rationale. For symbols that are partially correct or incorrect, both the positive (if any) and negative aspects of the symbol are described in the rationale, i.e. why the symbol (or some aspect of it) is a bad choice in some way. The rationale for the symbol may also list principles that the student has failed to apply by his choice of the symbol. Also associated with each symbol, the instructor inputs a numerical weight as to the importance of the existence of the symbol in the student's plan from the perspective of determining whether the student's concept of operations matches that of this stored plan. For example, central to the concept of operations of a particular instructor entered plan may be a frontal assault on the main enemy unit. So the attack arrow symbolizing that assault would have a very high weight. Other aspects such as SBF positions or protections of the flank (although tactically important) may be a lot less important in terms of determining that the student's plan is based on the same basic concept of operations. Also associated with each symbol or each plan or each scenario or the application as a whole is a quantitative description of how to rate the similarity of a student entered symbol and an instructor entered symbol when they partially match. For example the symbols existence in the general area, specific location and unit type may be of equal importance. If the student's unit type is only similar to the instructor's (e.g. mechanized infantry might be considered similar to an armor unit of the same size) then perhaps such a partial match would only receive half credit

Armed with this information, automatic AAR software can evaluate and debrief a student's plan using the following procedure. The student's plan is compared to each of the instructor-entered stored plans for the same scenario and a similarity score is calculated. The similarity score is the weighted average of the match of each symbol in the student's plan to the most closely corresponding symbol in the stored plan where the weights used are the ones entered by the instructor for each symbol. To illustrate this calculation, consider the very simple example below. The stored plan consists of two symbols, a tank platoon at a particular location and an attack symbol showing the tank platoon attacking along a certain route. The instructor has also specified the weights for each of these symbols and has also specified that in general when matching symbols, existence, specific location, and unit type are of equal importance (each 1/3) and that a tank unit type matches a mechanized infantry unit of the same size with a value of 0.5. (He might also have specified to further reduce the match by 50% if the size of the units are off by one echelon, but that is not used in this example). The student's plan includes an identical attack arrow symbol and so receives a matching score of 1.0. The student's other plan symbol is a platoon, located at a location close enough to the stored plan platoon symbol that it is considered to be exactly matching on location and of course it exists, but it mismatches in type since it is a mechanized infantry platoon and not a tank platoon. The matching score for that symbol is calculated as 1/3 + 1/3 + 0.5 x 1/3 = 5/6 = 0.8333. Then the weighted average of the two matching scores (1.0 and 0.8333) is calculated as shown.

Stored Plan		Student Plan		
Symbol	Weight	Symbol	Match	W x M
TankPlt	3	MIPlt	0.8333	2.5
Attack	22	Attack	1.0	22
Total:	25			24.5

Weighted Average = 24.5/25 = 0.98 = 98%

Once the similarity score is calculated for each of the scenario's stored plans, the plan with the highest score is chosen. There are two possibilities. Either the chosen plan is a correct one or one of the others. In the former case, the student receives just a confirmation that he has a good concept of operations and optionally receives credit for correctly applying all the principles associated with the high level rationale. Generally a link is also provided to the matching stored plan's rationale in case he wants confirmation or more detail as to why this concept is a good one. The software then reexamines the match between each of the student's plan symbols and the stored plan's and generates a small debriefing on each mismatch. For each symbol in the stored plan with no corresponding symbol the student is told about his missing symbol and receives the rationale for why the symbol is required (i.e. why the task is important). For each partially matching symbol, the student receives the rationale for the mismatching aspect. For example, if the student has chosen the wrong type of unit, he receives the rationale for why the correct unit type is chosen. He also receives confirmation that all of his matching symbols are correct (and optionally why). (In general, it is a good idea instructionally to give feedback on a student's correct actions so that he knows for sure that they were correct.)

In the case where the most similar plan is either partially correct or incorrect, automatically assembling the debriefing is more complicated. The high level rationale from the most similar plan is presented to the student along with the negative aspects of this concept of operations. Optionally the positive aspects could also be presented. Usually the pointer to an entirely correct plan would also be provided. The match between each of the student's plan symbols and the stored plan's is reexamined. When the stored symbol represents a correct one, matches and mismatches are handled as described in the previous paragraph. When the stored symbol represents an incorrect one, it is handled roughly in the opposite way. Failure to match it is not mentioned (the student didn't plan an incorrect task). If the student did match the stored incorrect symbol, he receives the rationale for why that symbol is incorrect.

Throughout the above procedure, the automatic software can accumulate the principles associated with the overall plan and those associated with the individual symbols into two categories - those that he has shown he can apply during tactical planning and those that he failed to apply. For the failed principles the automatic AAR software can present multimedia descriptions that explain them, if such files exist.

PLAN/CBR EVAL EXAMPLE

As an example of the above ideas, consider a simplified example from an actual automatic plan debriefer [Stottler & Pike 2002]. This scenario was itself adapted from an actual scenario used for training Armor company captains at Fort Knox. The situation is that an invasion is planned into a particular country. The landscape is flat, open, and covered with tall unharvested sugar cane with dirt roads criss-crossing the countryside frequently. A single US mechanized infantry company (located bottom most in Figure 5 (please ignore the arrows for now)) is tasked with seizing and holding an enemy airfield for a one hour period at which time they will be relieved by substantial airborne forces that will land at the seized air strip. They are to have seized possession by 9:00 am and hold the airport until 10:00 am. Hostilities will commence at 7:00 am. The airfield is defended by an enemy platoon (located leftmost in the figure) of mechanized infantry anticipated to offer only light resistance. Three enemy barracks, each housing a company of mechanized infantry are located in the general area of the airport (the three rightmost units in the figure). It is anticipated that once hostilities start that they will remain in place for at least the first hour, while they recover their officers and get organized.

One correct solution is shown in Figure 5. The instructor would create this plan consisting of 4 attack arrow symbols for use by the automatic debriefer. (All of the other symbols, including the friendly unit's initial position, were given to the student and therefore are not considered part of the plan). He would then input annotations consisting of the rationale for the concept and individual symbols. The entered rationale for the overall concept would point out that defeating the enemy forces will be easier if they can be defeated separately before they combine into a single large force. Additionally those forces are effectively immobilized for the first hour. And finally, to accomplish the objective, only the enemy vehicles have to be destroyed, because the enemy soldiers will not be able to walk to the air strip before 10:00 am.

The annotation for the first arrow would describe that given that the enemy will only be assured of remaining in their barracks for one hour, it is important to attack the vehicles at the closest barracks first. It might also describe the importance of taking the fastest route which happens to be the dirt road (not shown) under that attack arrow. Similarly the rationale for the second arrow would discuss the need to attack the unit closest to the site of the first attack, etc. The rationale for the final arrow would reference the commander's intent to seize the air strip. It would reference the location of the arrow as being the most direct route from the attack at the last barracks. The instructor would weight the first arrow the most, since this indicates that the student's concept is based on attacking at least some of the enemy units at their barracks in place

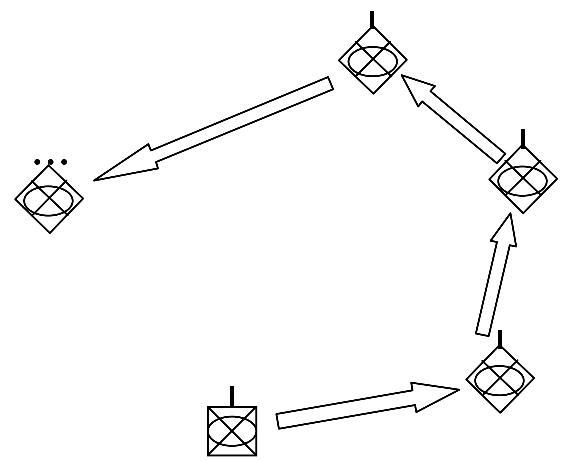

Figure 5. Scenario Setup and Correct Solution

A common incorrect solution that the instructors anticipate that many students would try is shown below and consists of one attack arrow. The rationale describing why this was a bad concept would point out that the terrain is not particularly defensible. While it will be relatively easy to overcome the air strip's defending platoon, once the enemy units housed in the nearby barracks combine they constitute an overwhelming force so that the student will not be able to meet the objective of holding the airport until 10:00. It would also contain a pointer to the correct solution. Because this plan has only one symbol, there are not significant annotations associated with that symbol.

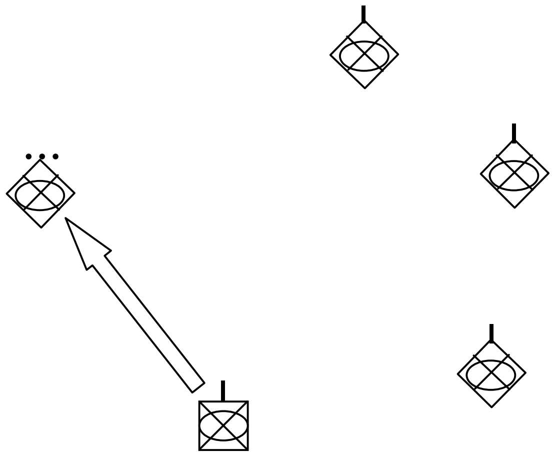

Figure 6. Common Incorrect Solution

Consider a different scenario from the same automatic plan debriefer illustrated below in Figure 7. A heavy mechanized infantry company is road marching when the lead unit (a tank platoon) observes an anti tank ditch across the road at a chokepoint. The terrain on either side of the ditch has a steep slope and is forested and impassable. The company's other assets include two mechanized infantry platoons and a combat engineer platoon.

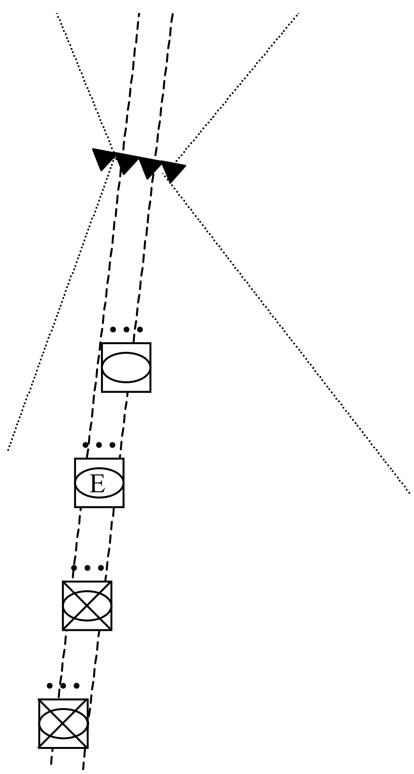

Figure 7. Chokepoint Scenario

A correct plan is shown below in Figure 8. The annotations an instructor would make for the overall concept would point out obstructions are only useful if they are overwatched and that the assumption should be made that there are enemy infantry units on the high forested ground on both sides of the obstruction. Therefore the overall concept is to dismount the two infantry platoons and have each destroy the likely enemy units then secure the far side of the obstruction. Then the combat engineer platoon can breech the obstruction. The instructor would place the highest weights on the two initial attack arrows, since their existence in the student's plan would show he understands the concept of needing to eliminate the likely overwatching enemy units.

The individual symbol annotations for those attack arrows would describe the need to attack the likely enemy positions and the choice of dismounted infantry as the best for assaulting into impassable forested terrain. (In another example there might be some discussion as to the choice of the size of the attacking force.) The annotations for the second pair of attack arrows would describe the need to secure the far side of the obstacle and that the dismounted infantry were both in the best location (after their initial attack) and the best type of unit to perform this mission. The annotations for the two lower SBF position symbols would describe the need to support the infantry attack with the fire power of the Bradley fighting vehicles. The annotations with the upper most SBF position would describe the need to cover the obstacle itself and the area immediately behind it with our own firepower and that the tank platoon was the best choice since as the point of the formation, they would likely have to resist the most enemy fire power. Finally the annotations with the crooked arrow would describe the need to breech the obstacle before it could be traversed and that the engineer platoon was the best unit for this task.

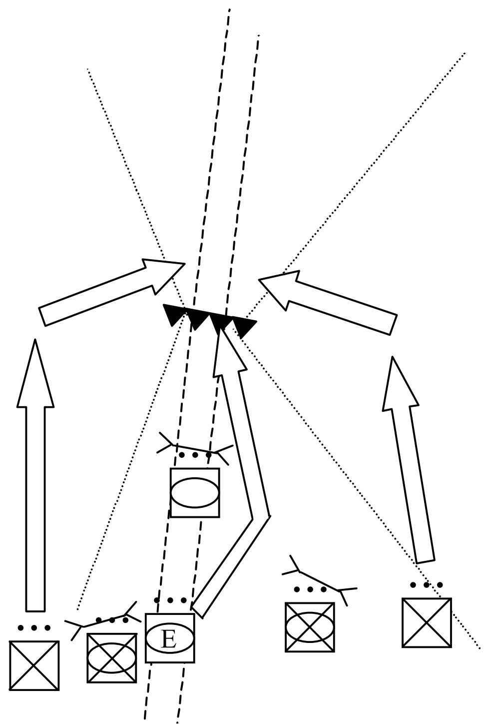

Figure 8, Choke Point Scenario Correct Solution

This comparison technique has also been used to produce automatic debriefings in military applications besides planning. For example, the Navy currently uses such an automatic system to train sonar technicians to analyze sonar images. The student's analysis is compared to the stored annotations and rationale of an expert to automatically evaluate the student's work and to provide a debriefing. It has also been applied to debrief the communication skills and tactical knowledge of AWACS Weapon Directors.

RESULTS AND LESSONS LEARNED FROM OPERATIONAL SYSTEMS

These techniques have been applied to several operational automatic AAR systems for both mission planning and execution training applications and several lessons have been learned. Most importantly, both the instructors and trainees view such systems very positively and actually realize greatly increased opportunities for tactical decision-making practice. For example, the TAO ITS allows ten times more tactical decision-making practice and an independently reviewer found an 80% extremely favorable rating from the trainees with only 1 respondent neutral and the balance favorable [Stottler & Vinkavich 2000].

Both techniques appear to dictate certain restrictions on the types of scenarios that should be developed for training so that the techniques will apply. However in practice this constraint seems to already have been applied to scenarios used currently for training, because instructors generally use existing scenarios as a basis for both types of automatic AAR systems (BTNs and plan comparison).

Similarly both techniques are really based on the idea that student actions to a situation (the order to plan for a specific operation or a situation encountered during mission execution) will fall into one of a few categories and that responses outside of these categories can be treated in generic manner. This turns out to be true, in practice, to a remarkable degree as evidenced by both the instructor and trainee acceptance.

Two comments are in order regarding the concept of non-programmer editing and creation BTNs. The first is that we have found in practice that tactical experts can develop their own BTNs, especially in pairs where one member is tactically knowledgeable (typically an older instructor) and the other is more computer savvy (typically a younger lower ranking officer). However for SME editing to be feasible requires some thought into the design of the first BTNs and to carefully select the set of conditions, events, and their parameters that the BTNs will reference. They should be well-named and described and developed from the perspective of the tactical experts, not from the perspective of a programmer.

Mission execution decision evaluation involving terrain will often involved scenario specific (or at least terrain specific) BTNs. This is because terrain analysis is still best done by using experts. Given tactical situations likely in a scenario, they can determine good locations (and seemingly good but actually bad locations) for certain types of tasks such as key terrain, good SBF positions, defensible positions, etc. These can be defined on a map of the terrain and referenced by name in the BTNs.

One of the benefits of the plan comparison debriefing method is the rich explanations that can be provided regarding the concept or individual symbols. [Schmitt 1994] has several good examples of detailed, rich explanations related to specific likely courses of action.

REFERENCES

Schmitt, J. (1994). *Mastering Tactics: A Tactical Decision Game Workbook* , US Marine Corps.

Stottler, R., Jensen, R. & Pike, W. (2002), "Adding an Intelligent Tutoring System to an Existing Training Simulation", I/ITSEC 2002.

Stottler, R., & Pike, W. (2002), "An Embedded Training Solution: FBCB2/Tactical Decision-Making Tutoring System", I/ITSEC 2002.

Stottler, R. & Vinkavich, M. (2000), "Tactical Action Officer Intelligent Tutoring System", I/ITSEC 2000.