DISTRIBUTED INTERACTIVE INTELLIGENT TUTORING SIMULATION

John Leddo, Ph.D.
Research Development Corporation
Herndon, VA 20171-3205

James Kolodziej
Research Development Corporation
Herndon, VA 20171-3205

Simulation has played a major role in military training. Distributed Interactive Simulation (DIS) allows multiple trainees to interact in real time on a common training problem. While DIS is a powerful training tool, a trainer is typically required to review trainee performance and make the appropriate teaching and remedial points. As training scales to larger and larger exercises, the trainer will naturally focus on general team performance at the expense of individual training needs. Intelligent tutoring systems (ITSs) have focused on providing instruction on a one-to-one basis. Integrating DIS and ITS technologies offer the opportunity to capitalize on the strengths of both: the ability to conduct large scale team exercises while providing each trainee with personalized instruction. The present paper reports a Phase II Small Business Innovation Research (SBIR) project, sponsored by the U.S. Army Simulation, Training and Instrumentation Command (STRICOM) in which a Distributed Interactive Intelligent Tutoring Simulation™ was developed to train Army Infantry squad and fire team leaders the skills they need to cooperatively perform military operations in urban terrain (MOUT).

ABOUT THE AUTHORS

Dr. John Leddo is President of Research Development Corporation. He earned his Ph.D. in Psychology from Yale University in 1988. Dr. Leddo specializes in developing training technologies along with the cognitive learning models upon which the technology is based.

Mr. James Kolodziej is Chief Technical Officer of Research Development Corporation. He holds an MS in Computer Science from George Mason University. Mr. Kolodziej leads RDC's technical team in developing intelligent tutoring system and intelligent agent technologies.

DISTRIBUTED INTERACTIVE INTELLIGENT TUTORING SIMULATION

John Leddo, Ph.D.
Research Development Corporation
Herndon, VA 20171-3205

James Kolodziej
Research Development Corporation
Herndon, VA 20171-3205

BACKGROUND

Simulation has long played a significant role in military training. Whether live, virtual or constructive, simulation is the primary mechanism by which soldiers receive practical training or conduct analysis to prepare for potential missions that they may be called upon to accomplish. The present paper focuses on the use of simulation as a practice tool rather than as a mission analysis tool.

As technology has improved emphasis has been placed on creating simulators with higher fidelity and networking simulators to support team problem solving. In the military, this is best exemplified by the creation of distributed interactive simulation (DIS). DIS supports training by allowing large numbers of trainees to work together in order to collaboratively solve a problem as they would in a real setting. This is perhaps the most critical innovation within the field of simulation as virtually all military problem solving exercises are collaborative activities. As technology improves, one can only expect to see a capability to provide even more comprehensive training exercises with greater numbers of participants.

As technology has increasingly enabled training exercises to increasingly approximate realism by means of DIS-based virtual simulations, a subtle tradeoff is being made. On the one hand, the increased realism provides greater transfer of problem solving skills to the real world tasks a soldier will perform. On the other hand, as simulation substitutes for instructor-based training, the learning benefits of having active assessment and feedback by a qualified instructor is sacrificed.

The present paper addresses a project we have worked on that tries to give the training community the best of both worlds: distributed interactive simulations for realistic team training and individualized instruction for each soldier based on his learning needs. To accomplish this, integrated intelligent tutoring system technology has been integrated with distributed simulation technology.

Intelligent tutoring systems (cf., Brna, Ohlsson and Pain, 1993; Greer, 1995) use artificial intelligence to place teaching mechanisms into a training system. An intelligent tutoring system (ITS) is defined by three characteristics: an expert model of how to solve a problem correctly, a model of what the trainee knows that can be compared to the expert model, and a pedagogic model that tells the system how to teach the trainee what an expert knows given what the trainee knows.

Each of these components is missing from typical technology-based simulation environments. For this reason a typical simulator can provide practice but not instruction. The contention of the present paper is by integrating the approaches offered by the ITS and simulation communities, an enhanced paradigm of realistic DIS scenarios, coupled with the instructional benefits of ITS technology can be created. We term this technology “distributed interactive intelligent tutoring simulation” or DIITS.

In order to create a successful DIITS environment, several technical hurdles must be overcome. This is so because the strengths of each technology are problematic for the other.

For example, simulation environments typically support open-ended behaviors. A trainee can execute any action allowable within the simulator at any time, regardless of whether or not the action is prudent. An ITS, on the other hand, is trying to assess the student. Because it is much easier for a computer to process well-defined linear event sequences, most ITS work has focused on domains where well defined linear procedures exist, such as algebra and basic computer programming.

In a simulation environment, there need not be a predetermined “correct” answer to the scenario being trained on. Indeed, in many real world problems, a “correct” answer may not be known. In such cases, the instructor tries to evaluate the quality of solution process rather than the outcome of that process. In a military problem this could include such things as proper use of reserves, establishing supply lines, etc. In the ITS world, the emphasis has been on domains where an objective “right answer” exists (e.g., mathematics) as computers are much better at dealing with well defined problems with computable solutions than they are at dealing with ill-defined problems with no computable solution.

As noted earlier, the heart of DIS is collaborative problem solving. However, the goal of an ITS is to assess and teach each student based on his individual needs. Because it is easier to do this process when only a single student is being trained, ITS researchers have focused on single student ITSs.

The present paper addresses how these technical issues were reconciled so that a DIITS technology could be developed that supports open-ended team problem solving behaviors while providing each team member with individual instruction. In addition to these technical issues, there were other issues that relate to both the simulation and ITS communities.

One of the major weaknesses in the ITS and simulation fields is that both technologies are costly to develop. They typically require highly skilled professionals to develop. Once developed they tend to be highly inflexible such that they often become quickly outdated and need to be rebuilt from scratch when updated. When such rebuilding occurs, there tends to be little reuse of the previous system’s technology in the newer version.

An argument can be made that if DIITS technology is to have a future, it must be made adaptable and generic enough so that the technology can be constructed cost-effectively, be modifiable and updatable by the user, and have considerable reuse across projects to maximize the value to the user for the dollars invested.

These additional objectives influenced the design of the present technology (which is discussed in the next section). The effectiveness of the present technology was demonstrated by building generic and modifiable technology that was extended as the project progressed and was transferred to another project.

THE DISTRIBUTED INTERACTIVE INTELLIGENT TUTORING SIMULATION TECHNOLOGY

The present technology was developed in partnership with the following members of the Ft. Benning Infantry community: the Directorate of Training (DOT), U.S. Army Infantry School (USAIS) and the Battle Lab. We are particularly grateful to Mr. David Reiss who was our point of contact at USAIS and who provided us with support in the form of subject matter experts and project reviews.

In working with the Infantry community, the Military Operations in Urban Terrain (MOUT) task of clearing a building was identified as a high training priority. A decision was made to focus on leadership training, first at the fire team leader level and then at the squad level.

In both cases, leaders perform two major functions: first, they are responsible for making decisions and directing the actions taken by their respective units; second, they are responsible for correcting mistakes made by members of the unit. Therefore, the training objectives that were established for the present technology had both of these responsibilities in mind.

Part of an ITS is the expert model of how to solve problems in the domain. Additionally, a simulation requires a model for computing events within the domain. Both of these are clearly linked. The simulator needs to record trainee actions to compute how events should unfold in the simulation. The ITS needs to record trainee actions to determine learning needs.

In order to build the domain model and the expert problem solving model, extensive knowledge engineering was conducted with subject matter experts supplied by USAIS. This was supplemented by observing live training exercises at the McKenna MOUT Site at Ft. Benning, conducting role playing exercises at RDC headquarters and by reading published doctrine.

These knowledge models formed the basis of the training technology. The next step was to build the actual system. This was done in phases. In the first year of our two year project, a one person trainer for the fire team leader was built. In the second year of the project, a second, networked trainer for the squad leader was added.

In developing the trainer for the fire team leader, the goal was to create an architecture that would be later scaleable to adding the squad leader. A second goal was for the technology to be as generic as possible so as to be modifiable when necessary and usable with simulators other than the one created for the present project.

In order to meet these objectives, a simulation-based intelligent tutoring system was constructed that had three components. The first was the MOUT simulator that allows a trainee to assume the role of a fire team leader and direct a four man fire team in the task of clearing a building. The floor plan used in the simulator is modeled after a building within the McKenna MOUT site.

The simulator creates virtual (but non-immersive) simulations of the inside of a building. World Toolkit was used to create these simulations, which run on a standard Pentium PC.

The trainee assumes the role of the fire team leader and directs his team by means of key strokes. Once the key stroke command is issued, movement is automatic. The fire team carries out the commands issued by the trainee, regardless of whether they are “correct”. Occasionally, for pedagogic purposes, a fire team member carries out a procedure incorrectly. At this point the trainee has the opportunity to correct the mistake, again using keyboard commands. If the trainee does make the correction, the mistake is not repeated. If s/he does not, then the system continues to cause the fire team member to repeat the mistake.

The second component of the system is the intelligent tutor. The intelligent tutor is responsible for assessing the trainee actions in the simulator, determining whether corrective instruction is needed and directing the simulator to provide such instruction when necessary.

As stated earlier, an intelligent tutoring system is comprised of three components: an expert problem solving model, a student model and a pedagogic model. The logic for how these components were constructed is based on the project objectives. First, one of the major challenges to integrating ITS technology with simulation technology is to allow the ITS to support the open-ended behaviors and ill-structured problems found in many simulation environments. Second, it is desirable to have generic technology that is, in principle, reusable for other training applications and not tied to the specific simulator or set of scenarios that simulator might run.

In order to accomplish these objectives, two innovations were made to the typical expert model found in many ITSs. First, in order to handle the rich MOUT problem solving domain and allow more open-ended behaviors, a richer knowledge representation framework than is typically found in many production rule-based expert systems was used. This knowledge model framework was based on previous empirical research (Leddo et al., 1990) on how experts solve problems.

This research shows that experts use a variety of problem solving approaches that are richer than a simple production rule or other single formalism process. As a result, an Integrated Knowledge Structure (or INKS) framework was created that blends scripts, production rules, semantic knowledge and mental models into a single formalism. INKS allows an expert model to process known problem solving sequences as a production rule system would, but also allows a system to use mental model to reason from first principles given the semantic information available in a situation.

For example, when a fire team moves down a hallway and approaches a door, there is a fairly routinized procedure for stacking, breaching the door and entering the room. In cases where multiple doors are present a decision must be made as to which room to clear first. In such cases, at least two decision making processes are possible. The first is to "hardcode" every possible permutation of how many doors there are, whether they are marked or unmarked (indicating that they are already cleared) and whether the doors are open or unopened (indicating a potential threat as an open door constitutes a potential line of fire from enemies within the room). The expert system could literally evaluate each of the antecedent conditions to determine which rule to fire and in doing so, which room to clear.

An alternative approach, which the INKS expert model allows is to reason from first principles. To accomplish this, the INKS knows about the goal of preserving the safety of the fire team. It knows that being in an enemy line of fire constitutes a safety threat. Therefore, when confronted with multiple rooms, the INKS can evaluate each room to determine which constitutes the greatest safety threat and then decide to clear that one first. By using this mental model approach to evaluating safety threats, the system is not required to have a preset rule to assess a trainee's decision, but can still make the assessment based on known goals and situational (semantic) features.

Being able to reason from first principles leads into the second objective discussed, namely creating an ITS that is generic. Having mental models that reason from first principles is a step in the right direction as this supports reasoning about general cases rather than hardcoded examples.

This train of thought was continued by building the INKS to be a generalized MOUT expert rather than one that was knowledgeable about the specific simulator, floor plan or scenarios used. This was accomplished by encoding the domain knowledge in generic terms such as moving down types of hallways rather than specific hallways, entering types of rooms rather than specific rooms, etc. By doing so, the system is able to reason about any scenario as long as it can make the determination about what type of situation it is in. This makes the ITS independent of the simulator.

A good analogy is be a human trainer. A human trainer is a generic domain expert. S/he can be brought into any training environment and act as a trainer because s/he knows about domain and can apply that knowledge to the specifics of the training environment because s/he can get, by observation, the relevant information about what learning objectives are being taught and what scenarios are being used to teach them. Before discussing how the ITS is getting this scenario information, the other two components of the ITS are briefly discussed: the student model and the pedagogic model.

The student model uses the same INKS used in the expert problem solving model. In this way, student actions are compared directly to the expert model and assessed accordingly.

For the pedagogic model, there are many options as there is no "one right way" to teach any subject area. Case-based reasoning was chosen as the pedagogic model for two reasons. First, it lends itself well to the scenario-based instruction that the simulation would use. Second, it supports individualized instruction as the student learning style itself can be treated as a case to which remedial instruction could be matched. In the present system, four forms of remedial instruction are used: showing trainees the consequences of their actions with logical scenario outcomes (e.g., a trainee fails to correct a fire team member action that places the fire team in danger. Therefore, the case-based reasoner causes a soldier to be killed by an enemy.); providing trainees with miniature scenarios to practice a faulty skill; providing trainees with an auditory explanation of what they did wrong; and providing trainees with a text-based explanation of what they did wrong. The case-based reasoner (CBR) cycles through these potential forms of remediation until it finds a form by which the trainee does not repeat the mistake being remediated. When this occurs, the CBR infers that it has matched the trainee's learning style.

Having discussed the simulator and the ITS, the third component of the technology is addressed. A generic INKS was created to serve as the expert problem solving model. This INKS operates as a human trainer does, inferring from the events in the simulation what type of scenario is being trained so that it can assess the trainee and determine whether remedial instruction is needed.

In order to accomplish this, real time feedback is needed as to what events are happening in the simulation and what actions are being taken by the trainee. To accomplish this, middleware, which is the third component of our technology, was created.

The middleware has two major components. First, in order to provide the INKS with information regarding events in the simulator, a semantic overlay of the floor plan used in the simulation was created. This semantic overlay contains such information as where the doors are, whether they are open or closed, how many locks and hinges they have, etc. Therefore, as the trainee moves through the simulation, the middleware can continually pass this information to the INKS. In essence the middleware acts as the “eyes and ears” of the INKS so that it can evaluate what type of situation the trainee is in and then use its expert model to determine appropriate actions the trainee should take. When the trainee does take an action, the middleware also passes this information to the INKS so that it can assess the trainee’s action against its expert model.

If the trainee’s action matches the expert model, then no remediation is provided. If remediation is needed, the CBR generates a requirement. This is passed to the middleware. Here is where the second major component of the middleware comes in. The middleware also contains information about simulator primitives so that it can cause the simulator to produce necessary remediation. The basic simulation-based ITS architecture is illustrated in Figure 1 below.

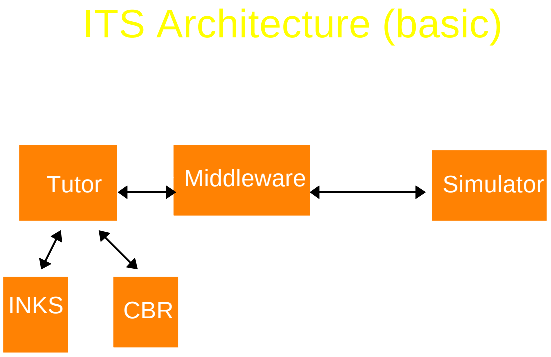

Figure 1: Basic ITS Architecture

In year 2 of the project, the goal was to add the squad leader to the training technology. As was the case in the first year of the project, the first step was to conduct knowledge engineering with domain experts, supplemented by published doctrine to gather the necessary domain knowledge.

However, the goal in year 2 was more than to simply build a second simulation-based ITS. The goal was to create a team trainer. Here, the challenges of creating a distributed problem solving environment while still preserving individualized instruction needed to be addressed. There was an additional technical challenge. The project goal was to create a DIITS for a squad leader and a fire team leader working together. However, a squad has two fire teams. Therefore, in order to preserve the realism of a two fire team squad, intelligent agent technology was used to play the role of the second fire team (this technology is discussed shortly).

Fortunately, the basic year 1 architecture supported these extensions. First, the MOUT simulator was reused in year 2 (the floor plan was extended to provide a starting hallway for the squad). In order to give the squad leader a separate perspective corresponding to what he would see, a duplicate simulator but with a camera (viewpoint) corresponding to what he would see was created. The squad leader was also provided with a sky view so that he can still watch the actions of the fire team after they disappear into a room. Finally, while the fire teams move as a unit, the squad leader is not tethered to either fire team but moves himself manually using key strokes. However, when he directs the movement of one of his fire teams, their movement is automatic as was the movement of the fire teams when directed by the fire team leader in the year 1 system.

The next step was to build the appropriate ITS for the squad leader. The key step was to build the squad leader expert model. This was done in the same format as the fire team INKS. The CBR was also updated to include the remedial instruction for the squad leader errors. By creating separate tutors for the fire team and squad leaders, an environment was created where each trainee had his own tutor that was responsible for providing him with personalized instruction based on his learning needs.

Once the simulator with the squad leader perspective and the squad leader ITS were created, middleware was constructed to link the two as was done in the fire team leader trainer. This middleware was essentially the same as for the fire team leader. The semantic overlay of the floor plan was identical. The main difference was to be able to pass the squad leader commands in the simulator to the squad leader ITS.

The technology, as described above, constitutes two separate ITSs, even though the second one was constructed far more cheaply than the first. However, the goal was to develop a team training environment so the two systems had to be linked. The linkage was provided through the middleware. Here, in each system, the middleware passed each trainee's simulator commands not only to his own ITS but to the other trainee's middleware. The other middleware then updated the second simulator so that the trainee would view the same events as his partner. To synchronize the two simulators, an internal simulation clock was created. Each event that was passed between the two middlewares was timestamped so that the receiving middleware could update its simulator in a way that would preserve the synchronization between the simulators. The synchronization between simulators is a standard DIS problem and the present paper does not claim to have made any innovation in this area.

The resulting technology was two Pentium PC computers that used a point to point connection. One computer had the squad leader version of the technology, while the other had the fire team leader version. Figure 2 below shows the architecture for the two person DIITS.

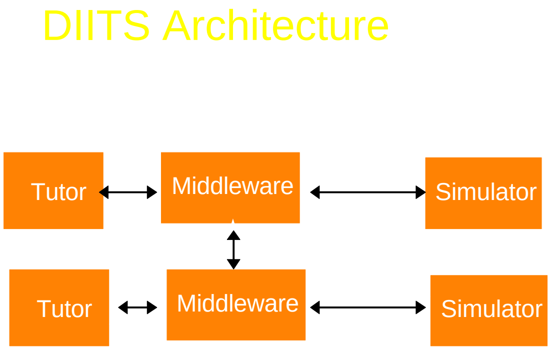

Figure 2: 2 Person DIITS Architecture

There is one final issue. The present technology is comprised of one squad leader trainer and one fire team leader trainer. However, a full squad has two fire teams. The second fire team was "played" by an intelligent agent.

One of the features of the present ITS technology is the expert problem solving model. This expert problem solving model evaluates the trainee by computing its own solution to the problem and evaluating the trainee against that solution. This feature of the expert model was used in order to create an intelligent agent that would do the same.

Therefore, when the squad leader issued a command that would ordinarily be carried out by the second fire team (fire team B), the expert model would generate an expected action on the part of that fire team. These commands were then automatically carried out. This enabled the agent for fire team B to respond to squad leader commands.

There were cases where the fire team leader would normally issue his own commands. In this case, the expert model would be receiving information, via the middleware, of what events were happening in the simulation. The expert model then computes what the fire team leader should command. In this case, rather than waiting for the trainee to issue a command, the expert model simply issues the command itself.

This ability for the expert model to operate either in assessment mode (when a real trainee is issuing commands) or in agent mode (to issue a command itself) created a unique feature of the technology. Specifically, the DIITS could not only support two person training, but also single person training in either a fire team or squad leader role. This was accomplished by having a toggle that transferred control of the fire team leader or squad leader from human to computer and back again.

DEMONSTRATION OF THE ROBUSTNESS OF THE TECHNOLOGY

A principal goal of the present project was to create a generic DIITS architecture that could be a model for rapid technology construction and software reuse. While a demonstration of this was not part of the original project plan, there was a serendipitous opportunity to provide such a demonstration when RDC received a Phase I Small Business Technology Transfer contract with the Army Research Institute to develop a virtual schoolhouse (see separate paper, entitled "The Virtual Schoolhouse", also submitted to these proceedings).

In this project, the goal was to transfer DIITS technology from the present project and intelligent agent technology from a DARPA-sponsored project to develop team training that could be delivered over a network such as the Internet.

Because virtual simulations require higher bandwidth to be transmitted over a network than constructive simulations, a decision was made to use a constructive simulation for that project. The entire functionality of the present project was duplicated in the form of a networked constructive simulation for the ARI project. To accomplish this, both squad and fire team INKS' were reused in their entirety and enhanced them and the middleware was reused in its entirety. This was done by creating a constructive version of the same virtual simulation such that the primitives and semantic overlay provided in the middleware also mapped to the constructive simulation as well as the virtual one.

Essentially, then, the present project technology was completely reused and even enhanced (additional knowledge was added to the INKS in this second project), except for the virtual simulator. However, between the two projects it was demonstrated that the same ITS technology could be applied to two simulators (a constructive and virtual one) without changing either the ITS or the middleware and the same simulator could be applied to two different ITSs (a fire team leader and a squad leader) without changing the simulator (except for the camera angle) or the middle ware.

There was a further test of the robustness of the present technology. A scenario editor was created that allowed a user to enter his own scenarios by manipulating certain parameters (e.g., number and location of enemies, their level of training and whether they were combatants, number and location of civilians). As stated earlier, the present technology allows the computer run the role of the fire team leader or the squad leader because it processed scenario information in real time an made its own decisions regarding what actions to take.

In this case, numerous demonstrations of our technology were given. Each time, observers were allowed to create their own scenarios and have the expert model run the simulation. There were no cases where the scenario created by an observer "broke" the system or created an unexpected event. This demonstrates the robustness of our expert knowledge model, which is the heart of our technology.

It should be noted again that the virtual schoolhouse system completely duplicated the functionality of the present system. With the addition of the scenario editor, it actually had an additional feature that the present technology does not have. What makes this accomplishment even more noteworthy is that the virtual schoolhouse project was developed using Phase I level resources which are about one-sixth that of a Phase II development effort. This demonstrates how such technology, if carefully constructed, can help leverage future development efforts such that the technology can be replicated and enhanced at considerable savings.

CONCLUSIONS

The present project demonstrates the feasibility of integrating intelligent tutoring system and DIS technologies. It is argued that the value of such integration is by maintaining the team training spirit of today's DIS technology, while providing each participating soldier with the opportunity to receive individualized instruction based on his learning needs.

Further, the present project demonstrated an approach for creating such technology that makes it extendible and reusable at a fraction of the original development costs. This can be considered an important feature as it suggests that such functionality could be added to today's (and tomorrow's) simulator development efforts at a modest cost.

The goal in this project, then, has been to demonstrate the enhancements that could be made to DIS technology by integrating ITS technology and that these enhancements could be done in a cost-effective manner. By achieving these goals, it is hoped that a paradigm shift can be effected in the way simulation-based training is conducted. The authors' vision is that future training technology will be a marriage of simulation and intelligent tutoring systems, creating distributed interactive intelligent tutoring simulations.

REFERENCES

Brna, Ohlsson, S., & Pain, H. (1993) (Eds.) Proceeding of Artificial Intelligent in Education '93. Charlottesville, VA: Association for the Advancement of Computing in Education.

Greer, J. (1995) (Ed.) Proceeding of Artificial Intelligent in Education '95. Charlottesville, VA: Association for the Advancement of Computing in Education.

Leddo, J., Cohen, M.S., O'Connor, M.F., Bresnick, T.A., and Marvin, F.F. (1990). Integrated knowledge elicitation and representation framework (Technical Report 90-3). Reston, VA: Decision Science Consortium, Inc.