Use Cases, Requirements and a Prototype Standard for an Intelligent Tutoring System (ITS)/Simulation Interoperability Standard (I/SIS)

Richard H. Stottler and Robert Richards, Ph.D.
Stottler Henke Associates, Inc.
951 Mariners Island Blvd., Suite 360
San Mateo, CA 94404
650-931-2700
stottler@stottlerhenke.com

Brian Spaulding
MAK Technologies, Inc.
10 Fawcett St
Cambridge, MA 02138
617-876-8085 x112
bspaulding@mak.com

Keywords:
Intelligent Tutoring System (ITS), Simulation, Interoperability, Standard

ABSTRACT: *Intelligent Tutoring Systems (ITSs) are being applied to an increasing proportion of military training. Most ITSs are interfaced to simulations, usually involving real-time tactical scenarios. The simulation and ITS are generally developed by different companies at different times. This makes interfacing the ITS and the simulation problematic, experience has shown. An industry-wide interoperability standard would enable different vendors of simulations and ITSs to easily integrate their products, save time and money, and increase the training value of simulation exercises. The Intelligent Tutoring System Interoperability Study Group (ITSI SG) was formed to study the issues associated with an ITS/Simulation Interoperability Standard (I/SIS).*

This paper describes the results of the ITSI SG meeting held at the Fall, '04 SIW in Orlando which resulted in a refined set of use cases and requirements and a prototype of the I/SIS. They are presented here for comment and discussion and follow up on the paper, "Requirements of an Intelligent Tutoring System (ITS)/Simulation Interoperability Standard (I/SIS)" presented at the Fall, '04 SIW [1].

Four primary use cases were identified – tactical decision-making training, equipment operations and maintenance training, ITS-centered training systems, and simulation-centered training systems – along with combinations of them. Several requirements were identified and sorted by use case and level. The Study Group decided that it was important to have both a minimal level I/SIS that simulation developers could easily meet (Level 1) and would allow a reasonable integration and level of functionality and a higher capability level (Level 2) that provided a much better integration and supported a much fuller range of desired capabilities. A small number of optional levels were also identified to support very specific capabilities, such as an integrated scenario editor. The requirements include data from the simulation needed by the ITS, such as the trainee's actions for automatic evaluation; facilities needed by the ITS, such as an avenue to present feedback and other information to the trainee through the simulation; and facilities needed by the simulation, such as automatic trainee evaluation, on request.

The prototype standard allows for data transfer through either the Distributed Interactive Simulation (DIS) or High Level Architecture (HLA) protocols at the simulation developer's discretion. TCP-IP sockets are also being considered. The XML Battle Management Language (XBML) would be used to format tactical orders. Other data needed by the ITS or simulation would be in XML format.

This paper will present examples of each of the use cases and requirements from actual Simulation-ITS integration projects as well as examples of how the standard would have applied had it already existed. This paper supports the ITSI SG by promoting discussion of I/SIS use cases, requirements, and prototype standard with an audience larger than the study group.

1. Introduction/Motivation

In order to evaluate a trainee's actions and decisions, the ITS needs access to information about the execution of those actions. It may need a mechanism within the simulation to present real-time and debriefing feedback. The ITS may also need a mechanism to start the simulation with a specific scenario and potentially control some aspects of the scenario. Standards such as DIS and HLA (with various Federation Object Models (FOMs)) exist to support interoperability of simulations. These allow simulations written by different vendors at different times to interoperate. ITSs are often forced to use these standards as a basis for an interface; however, in the current form, their object models are inadequate and must usually be extended. Their primary shortcoming is that they were designed to export the data from one simulation to another primarily for rendering and calculations. This is what is externally observable for a given platform (position, velocity, sensor states, fire events, damage levels, etc.). However, a trainee's actions include data that is not externally observable such as actions performed on the equipment or in software that don't have an immediate externally observable effect or internal communications including text, audio, graphical plans, orders, etc. These are not handled by the common simulation protocols. Additionally, no standard mechanism exists for sending feedback to students through the simulation or for performing the control measures mentioned above other than some generic messaging packets and trigger mechanisms. A well-integrated application would also have a mechanism for transferring instructor entered information from the simulation's scenario editor to the ITS for use in evaluating the student's actions. A further benefit of I/SIS is that a human instructor needs access to the same kind of information and control as an ITS. Thus, I/SIS also will facilitate integration of simulations with instructor stations from different vendors, greatly reducing the cost of instructor stations while ensuring the needed instructor capabilities will exist.

2. Use Cases

Four important use cases were identified to cover the majority of military training using simulations and ITSs in concert. These were: Tactical Decision-Making training (TDM), Equipment Operations and Maintenance training (EOM), ITS-Centered training systems (IC), and Simulation-Centered training systems (SC). TDM training refers to training and practice in making decisions in a tactical situation. It is primarily directed toward deciding what to do. Equipment Operations training refers to training on the use of equipment or software to perform some task. It is primarily directed toward knowing how to do something, given that what to do has already been decided. Equipment Maintenance training, since it is directed toward tasks (troubleshooting, repair, preventive maintenance) not related to a tactical situation, doesn't involve TDM and requires the same type of ITS/simulation integration as operations. The EOM use case covers most miscellaneous training systems not related to the interaction of friendly and enemy units. For example, a medical ITS, a Navy sonar image analysis ITS and a counter-terrorism intelligence analysis ITS all fall under EOM.

IC training systems are ones in which the ITS is primarily in control of the combined application. The student or team typically starts training and logs on through the ITS. The ITS models the students and/or teams at least regarding the skills and knowledge that they have mastered, decides on the next instructional events, including which scenario is appropriate when the next event is executed in the tactical simulation, evaluates the performance of the students or teams, provides hints and real-time feedback, provides after action review (a scenario debriefing which may be interactive), and formulates and executes a remedial plan. The simulation provides the ITS with a practice and testing platform.

In SC training systems, the ITS is an accessory available to support and enhance the training objectives of the simulation. This support can include real-time feedback through an interactive dialog with the trainee, where the ITS evaluates performance during the scenario, and/or has an evaluation engine, that executes at the completion of the entire simulation or at pre-defined stages within the simulation. In all cases, the interaction with the ITS should be transparent and the trainee interacts with it as another component of the simulation-based trainer itself, rather than a separate item. The purpose of the ITS is to augment the basic functionality of the simulation-based trainer with intelligent evaluation, hinting, and debriefing capabilities and to provide a more effective training environment. However, the trainee starts and logs on to training through the simulation and should not need to start a separate application and, ideally, the ITS is seamlessly embedded and consistent with the GUI of the application itself.

These four use cases form two pairs each corresponding to a dimension. One dimension is the type of training – TDM or EOM. The other dimension is the type of system – IC or SC. A specific ITS/Simulation Integration covered by I/SIS will be at least one of each dimension. In other words, a specific I/SIS training application will be TDM or EOM (or possibly both) and IC or SC (or possibly include modes for both). As implied in the previous sentence, systems may include both use cases of the same dimension. In fact, a specific simulation-based

ITS might cover all 4 use cases. An example would be an ITS effort currently under way. An embedded ITS was developed for the Future Combat System (FCS) interfaced to an FCS vehicle prototype which teaches both platoon level reconnaissance TDM and how to use the robotic interface software. The current version is primarily SC, but the notion of having an IC mode has also been considered. If this occurs, all 4 use cases would be covered by the one training system. [2]

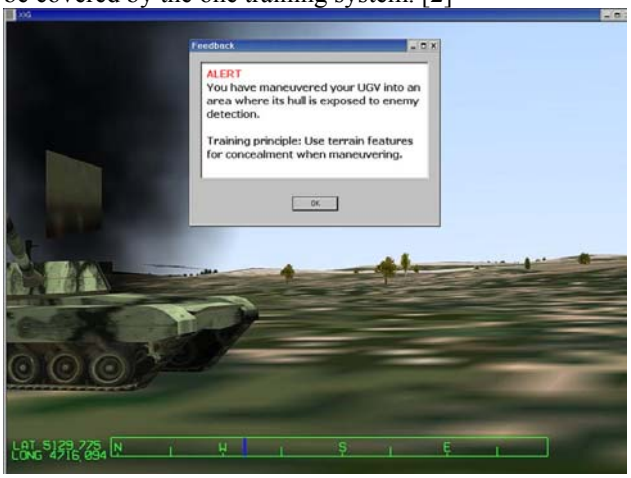

Figure 1. Embedded FCS ITS Prototype

3. Examples

Battle Command 2010 (BC-2010) is a PC-based, military tactical trainer that allows commanders and their staff officers to practice both their planning and execution skills within a compelling simulated environment. It was designed to support Army battalion and brigade commands in preparing operation orders.

BC-2010 helps commanders develop warfighting skills by allowing them to plan the battle, fight the battle and review the battle. At the start of training, trainees produce all graphical and text-based products to support their military decision making process. During this process, trainees collaborate using shared graphical overlays and planning documents. When trainees are ready, they can activate the simulation and fight their plan against other players or a computer-directed enemy. During the exercise, trainees can work together to revise the plan and issue changes to subordinate unit commanders. At the end of the execution, BC-2010 provides charts and tracking information to determine the success of the battle plan, as well as a full recording of the exercise as part of an After Action Review (AAR).

The BC-2010 ITS Integration Prototype is an example of the Tactical Decision-Making (TDM) and Simulation-Centered (SC) ITS interface use cases. Under a PEO-STRI funded project, MÄK Technologies and Stottler-Henke created an initial proof-of-concept prototype to demonstrate the linkage of an ITS with an HLA-based simulation, BC-2010. For the prototype, the ITS and BC-2010 existed as separate applications, with a separate window containing buttons to evaluate the user developed plan and evaluate the results of the execution. Although this is not the optimal approach for integration, it was done as a first step due to limited funding and time and did serve to demonstrate the additional value of incorporating ITS functionality.

To use the combined application, the trainee followed a deliberate planning process and created the tactical graphics, unit positions, and planning documents, then pressed a button to receive feedback from the ITS about the produced plan. Based on the feedback, the trainee could then make any necessary changes to the plan and re-evaluate. When the trainee (and the ITS) were satisfied with the plan, execution could be started. The trainee then executed the plan by tasking units according to the developed plan in an attempt to accomplish the specified mission objectives. When the execution was completed, the trainee pressed a button and reviewed the performance evaluation produced by the ITS.

The Intelligent Flight Trainer (IFT) teaches introductory helicopter pilots in the context of exercises performed in MicroSoft Flight simulator (MSFS) following the instructional model used by instructor pilots of teaching in the context of simulated flight exercises. Trainees are assigned flight exercises based on mastery and personality attributes. Exercises are preceded by pre-flight briefings, and followed by detailed after-action reviews. The after-action reviews include pointers to remedial material; however, most of the training happens in the context of exercises. Trainees are coached through the exercises to varying degrees based on their expertise level.

The IFT is an example of the EOM and IC use cases. To begin an exercise, IFT instructs the student to select a specific flight that is geared for the chosen exercise. The student uses the simulator facilities to start the flight. Once the flight starts, the student is in control of the helicopter. The tutor provides (spoken) instructions whenever it decides the student needs some help. The tutor decides and informs the student when the exercise is done.

Real-time coaching feedback during the exercise is derived from a description of the procedure the student should carry out to recover from out of nominal conditions (e.g., if the student lost altitude, use a recover altitude procedure to coach the student). Authors can "draw" such procedures as flow charts in a graphical environment for execution during the simulation.

The tutor provides the following kinds of coaching depending on the expertise level of the student. Advanced students get limited coaching, which is often restricted to alerting the student about events and helicopter conditions that need attention. Novice students, on the other hand, get hands on coaching in the form of specific instructions on what they should be doing with the controls. Coaching takes the form of verbal, spoken instructions. In addition, the tutor may provide help in the form of **visual cues**. In the current version of IFT, the tutor can flash relevant instruments to guide the student in using instruments to understand the state of the helicopter and determine corrective actions. For example, if the student is climbing too fast, the tutor will flash the *climb-rate indicator* in order to draw the student's attention to his rate of climb. The tutor stops the simulation whenever the student loses control of the helicopter. Losing control of the helicopter happens when the helicopter's parameters are outside the exercise's specified range, which is specified while defining the exercise using the authoring tool. Usually these parameters define obvious out of control conditions: the helicopter is about to crash, the helicopter is rolling, etc. Other out of control conditions are less obvious: the helicopter deviated too much from the exercise's targeted heading; the helicopter is out of the altitude range specified for the exercise; etc. The tutor will explain why the helicopter is out of control, as well as how to correct the situation.

The tutor provides an after-action review once the exercise is completed. This feedback is given in two forms: an exercise performance summary and a replay of the exercise. The postbrief is a typical exercise performance summary that shows the following results: (i) the three best things that were done well; (ii) improvements (if any) noticed in the student's flying skills, and (iii) three worst things done during the exercise. Hyperlinks are provided for the student to review those flying skills or principles that need the most improvement. In all cases, the student model is used to filter the feedback that is provided to the student by not including things the student already knows (e.g., pointing out things done well only if the student has not mastered them before the exercise) or things that are not usual problems (e.g., advanced students may lose altitude during an exercise although they, in general, know how to maintain altitude).

4. Requirements

One of the main goals of the study group was to make sure that the standard was practical in the sense of being straight-forward for simulations developers to meet. It was therefore decided to have two main levels of requirements and components of the standard. The first level, Level 1 (L1), should be very straight-forward for simulation developers, while still supporting most of the capabilities that the combined ITS/simulation training system should have. Level 2 (L2) in contrast supports almost all of the capabilities that such a training system should have. A number of optional levels for requirements and parts of the standard were also identified, corresponding to optional capabilities of the combined system. The optional level to allow for the ITS to provide feedback using the elements of the simulation's user interface is called LSUI. The optional level to provide integration of the simulation's scenario editor with the scenario information editor of the ITS is called LCSE (for coordinated scenario entry). The optional level for the simulation to allow the ITS to drive a replay capability is called LIDR (for ITS driven replay). Elements of the requirements and the standard itself are labeled with both the use case and level to be as clear as possible what is required to accomplish different sets of capabilities. There is a fair amount of overlap between the use cases so if neither use case is mentioned, the requirement or component of the standard is needed for both. Also, Level 2 implicitly includes all of Level 1.

The ITS requires access to the trainee's actions. For TDM, the Level 1 requirement (TDML1) is to provide externally observable data (such as that available through DIS or the HLA Real-time Platform Reference (RPR) FOM) to the ITS. The ITS will have to infer the student's actions from the actions of the tactical platforms, however this is often quite satisfactory. For EOM or TDML2 access to the internal actions (software or equipment actions, repair activities, communications such as plans, orders, audio, etc) is required.

The ITS also requires the context in which the student is making his decision. For TDML1 this will also be externally observable data, such as entity positions, fire events, etc. For TDML2, the student's immediate context will be provided by the simulation. This includes information dynamically displayed in the student's view. This would include what entities are visible in the student's current 3-D view, what vehicles are being displayed on the student's 2-D map, the values of various instruments, communications received, etc. EOM, both levels, also requires the values of various instruments and other information being displayed to the student.

A Level 1 requirement is for the simulation to present real-time and after action review information, both text and graphics, from the ITS to the student. A Level 2 requirement is for the simulation to provide interactivity with the student for the ITS. Instead of just presenting text and graphics, the ITS can ask questions and get the answers back, through facilities provided by the simulation. An optional level (LSUI) requirement enables the ITS to send instructions to the simulation to partially

control its user interface. For TDM this would be the ability for the ITS to highlight entities, units, or areas in a 2-D or 3-D display as well as allow students to answer the ITS's questions by selecting entities, units, or areas in the simulation's displays. For EOM, this would allow ITS highlighting or student selection of instruments, controls, equipment parts, tools, etc.

An IC requirement is for the ITS to be able to start the simulation in a specific scenario. The corresponding SC requirement is for the simulation to start the ITS, inform it of the specific scenario being run, and select specific evaluations, types of coaching, and debriefing that the ITS should perform. An ICL2 requirement provides for ITS control of simulation finish, pause, resume, reset, After Action Review (AAR) start, and various elements of the scenario including entities and units, controls, environment, and equipment malfunctions and inputs. During the debrief, the LIDR requirement allows the ITS to direct the simulation to replay various time segments from various perspectives.

An L2 requirement is for the simulation's scenario files to be in a standard format, for access by the ITS. Additionally, for TDM or EOML2, the ITS will access additional information associated with the scenario for evaluation and feedback purposes. Much of this information can be input by instructors at the same time that they are creating the scenario for the simulation. This could occur with separate editors, or optionally (LCSE), with integrated editors.

ITSs typically model the mastery and other aspects of specific students. This requires that they know which specific student or team of students is currently performing. Most have a logon process for students using a student ID and password. In the SCL2 case, these will have to be received by the simulation and passed on to the ITS. Additionally some simulations require a user ID and password logon process. In the ICL2 case these will have to be passed to the simulation. In both cases, the receiving component will have to notify the sending component that the either the user ID and password were accepted or that they were incorrect. In the case of team training, the SCL2 requirement also includes specifying for each student ID, that student's role within the team, assigned equipment, and other team context (such as call sign).

5. Preliminary Suggested Prototype Standard

All of the L1 and L2 requirements can be grouped into two categories – the need to move information between the Simulation system and the ITS and the need for the ITS or simulation to provide requested services to the other. This latter category also includes moving information because the Simulation or ITS has to make its request for services known to the other. The ITSI Study Group extensively discussed the most appropriate mechanism for this information transfer. Both HLA and DIS and other basic mechanisms were discussed. HLA and DIS were both considered serious contenders. Ultimately, it was decided that the standard should define the content and form of the information to be transferred but leave the selection of which mechanism option, whether DIS or HLA, up to the simulation developer. TCP-IP sockets are also being considered. This leaves the following architecture for the standard.

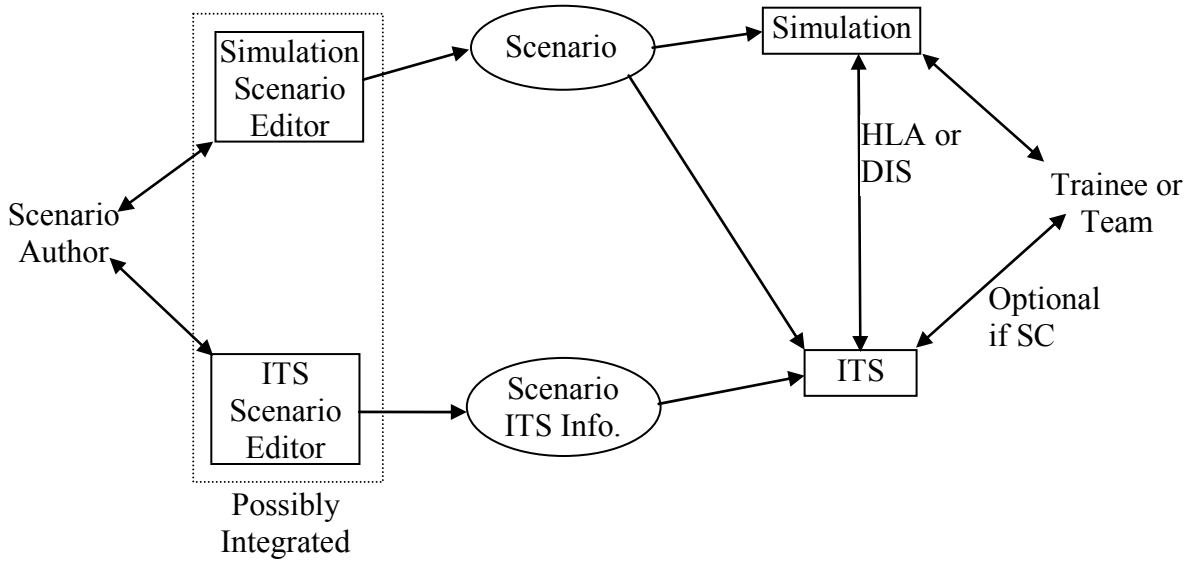

Figure 2 I/SIS Architecture

The trainee or team may or may not (in the SC case) interact directly with the ITS. They will definitely interact with the simulation through its user interface to perform in the simulated scenarios. The Simulation and ITS transmit information to each other, including requests for services, using HLA or DIS. In the case of HLA, this actually occurs through the Real Time Infrastructure (RTI), not shown. The scenario author interacts with the simulation scenario editor to create scenarios to be run in the simulation. For Level 2, tactical portions of the scenario are represented in the Military Simulation Definition Language (MSDL). Non tactical portions (or for the EOM case) are stored as XML files. The author also defines additional scenario related information for use by the ITS for evaluation, coaching, and debriefing purposes. The editor for this ITS related scenario information may (LCSE) or may not be integrated with the Simulation Scenario Editor. In addition to using the Scenario ITS Information, the ITS also accesses the Scenario used by the simulation.

If DIS is the chosen protocol, the information generally available from the simulation via DIS would still be transmitted via DIS. This corresponds to externally observable information and meets the requirements for TDML1. The other information, described further below, would be sent via Experimental PDUs. If HLA is the chosen protocol, the information generally available from the simulation in the RPR FOM would still be transmitted via HLA. This also corresponds to TDML1. Additionally, an I/SIS FOM extension will have been created that could be added to RPR FOM. This extension would encapsulate the additional details needed for ITS-Simulation communication. One specific additional Interaction to be created in the I/SIS FOM extension will be called Simulation Data and will be used in exactly the same way as the Experimental PDU, above. Then both the simulation and ITS would be Federates of the same Federation. This would not affect other Federates that were not aware of the I/SIS extension since they would have the choice of which Objects/Interactions to Publish or Subscribe. The decision to represent the additional data identically in either the DIS or HLA case was a conscious one because of the obvious benefits of one representation instead of two. A logical alternative for the HLA case would be to define the additional attributes in the I/SIS FOM instead of XML in one FOM class (the Simulation Data interaction). This would more naturally utilize HLA's capabilities at the cost of having different representations for HLA and DIS.

In either the DIS or HLA case, the additional information would be represented as XML at a minimum and with an XML based standard when available. Defining this extra information would be a good start. The HLA IEEE 1516 standard uses XML as its FOM definition language, so it would be easy to add the data to a FOM like RPR. Also, it would be a straight-forward process to translate the XML format to the HLA 1.3 OMT format. As discussed in requirements there are several different types of information to be transmitted.

TDML2 information of the student's actions includes orders which would be represented using the XML-based standard XBML [6, 7]. When those orders reference graphical plans, they would be represented using similar objects models as contained within existing FOMs like the Naval Training Meta-FOM (NTMF) [3] or the C4I Reference FOM [5]. Student actions that involve software or equipment actions are covered under the EOM student actions below. Communications that are not orders, will be represented depending on what they are. If they are C4I types of messages (such as to maintain a common operational picture) components of the NTMF or C4I Reference FOMs will be used. If they are text that does not fit this definition, they will be represented as text within an XML wrapper that describes whatever formatted data is associated with the text, such as the communication channel and the intended recipient. Similarly, audio communications are saved to a globally accessible file and an XML message that contains the file name and related attributes is created and sent to the ITS. For example, both DIS and HLA can be used to transmit digital audio corresponding to radio transmissions between team members. These would be stored in a file and the file name transmitted in XML via HLA or DIS to the ITS. Tactical context is partly provided by the list of entities and units shown in each display formatted in XML. Additional context includes communications previously received by the student which are represented in the same ways as communications from the student as described above. Context provided by instrument values are covered in EOM context below.

EOM information of the student's actions and context is represented by XML formatted lists of controls actions and instrument values. Displays that display multiple items have a list instead of a single value associated with them. If instruments do not always reflect reality or if some significant simulation state is not available from an instrument display, then simulation state information must also be sent via XML messages. For some applications, most notably pilot training, the real-time performance of a large number of XML formatted messages has to be studied, given the relative inefficiency of XML coding. The study may determine that compressed XML may be acceptable or that a different, more compact and easily decoded format may be needed for some applications. If so, the simulation developer will have to take extra care to clearly define the format and provide numerous examples.

Service requests are handled just like the other

information transfers, as XML formatted messages embedded in DIS PDUs or HLA objects. Both DIS and HLA have Action Request (and Action Response) messages, so these will be used to send the XML formatted service requests. Service requests have fields for the type of service request, the sender, and the intended recipient (the presumed servicer of the request) along with fields for the necessary information. The Level I requirement for the simulation to present feedback to the student from the ITS is provided by the Feedback service request type. It includes the URL of an html page of text and graphics to be displayed to the student. The L2 requirement to provide interactivity is provided by the Interactive Feedback service request type. It is otherwise formatted identically except that the html page is designed to return a value, either selected or entered, that the simulation should extract and send to the ITS as a service response type of XML message in an Action Response message. Multiple levels of interactivity are realized by the ITS receiving service response messages and responding with appropriate additional Interactive Feedback service request messages.

Both HLA and DIS include a corresponding set of Simulation Management (SIMAN) capabilities which I/SIS will take advantage of. However, these do not include starting the simulation application. The ICL1 requirement for the ITS to be able to start the simulation in a specific scenario will be handled by the ability of the simulation application to be run from a command line and to accept as its first argument on that command line the name of the scenario file. The simulation application starts up and loads the scenario, but does not begin the scenario. Analogously, the SCL1 requirement for the simulation to start the ITS in a specific scenario will be handled by the ITS application being able to be run from a command line and to accept as its first argument the name of the simulation's scenario file. The ITS must map from this file name to its own files for the scenario. By receiving the simulation's scenario file, the ITS has the opportunity to extract scenario information from it. The L2 requirements for scenario Finish and Pause and Start and Resume, are handled by the DIS Stop/Freeze and Start/Resume PDUs, respectively and the corresponding HLA interactions. The IC use case requires the ITS to emit these and the simulation to accept them while the SC use case requires the simulation to emit and the ITS to accept them. The ICL2 requirements for simulation reset, new scenario load, and AAR start are handled by the ITS sending Reset, Load Scenario, and Start AAR service requests through Action Request messages to the simulation which accepts and responds by resetting the scenario to the beginning, purging any existing scenario and loading the specified scenario file, and starting the AAR phase of training, respectively.

For SCL2, the ITS must honor a variety of related service request types sent through Action Request messages including Add Evaluation, Delete Evaluation, Run Evaluations Continuously, Run Evaluations Once, Run Specific Evaluation <Continuously | Once>, Add Coaching, Delete Coaching, Run Coaching <Continuously | Once>, Run Specific Coaching <Continuously | Once>, Add Debrief, Delete Debrief, and Run Debriefs to select and run automatic evaluations which posts its results as service response types in Action Response messages; select and run different types of automatic real-time coaching and hinting provided by the ITS which posts the resulting feedback as html pages in either service responses or in Feedback service request messages to the simulation, depending on a parameter in the Run request; select and run different debrief construction agents whose output is handled similarly to real-time coaching's.

The L2 requirement to allow control of various elements of the scenario including entities and units, controls, environment, and equipment malfunctions and inputs is handled in HLA by the ITS requesting ownership of specific attributes provided by the simulation as controls. The simulation divests ownership in those attributes but maintains control over the physical modeling. For example, the simulation might provide an attribute for a piece of equipment such as Status with possible values of Functioning, Total Failure, Partial Failure Mode 1, and Partial Failure Mode 2, which the ITS could use to introduce specific types of malfunctions, but the simulation would retain the modeling responsibilities for that equipment and it's interactions with other aspects of the simulation. Similarly, the simulation could provide for an enemy tank entity the attributes of Heading, Desired Speed, Aim Point, and Fire Trigger, which the ITS could use to control the tank but the calculations of actual speed, position, and shell trajectories would all be handled by the simulation. In the DIS case, Set Data and Data Query PDUs are used for this same purpose. Note that this L2 mechanism is similar in purpose but different in form to the mechanism described near the top of this subsection. That mechanism defined a specific standard for transmitting data needed by the ITS for evaluation (as XML in an Experimental PDU or Simulation Data object). The need to have access to internal information for controlled entities (which is also data transferred from the simulation to the ITS) in order for the ITS to exert intelligent control over them is more esoteric and therefore more acceptably left to the simulation developer's discretion. Unifying these mechanisms is an alternate option at the cost of complicating some L1 implementations.

For the L2 requirements, student IDs will be Universally Unique Identifiers (UUIDs). For IC and SC control the

simulation and ITS, respectively, will accept Logon service requests that include a password string and respond with an Action Response containing "Accepted" or an error message. For team training SCL2, the simulation will send an Assign Team Member service request containing the student's UUID, role, assigned equipment, etc. which the ITS will accept and note.

Some training simulations allow the recorded student performance to be replayed during an AAR session which follows the simulated scenario. The optional capability provided by the simulation to allow ITS-Driven Replay (LIDR) is handled by a series of service requests sent via Action Request messages. These are Set Time, Set Perspective <display element> <new perspective>, Play <real-time multiplier>, and Freeze, which are accepted by the simulation, while in AAR mode. Set Time sets the scenario time for the replay. The simulation's student interface may have a number of different displays, such as a 3-D out the window display and 2-D dynamic map for TDM training or a 3-D virtual world and instrument readings for EOM. The assumption is that during replay, all displays are synchronized to the replay time. However some displays may include the notion of a perspective. During replay, the 3-D view may be from out the window of the student's vehicle, from the vantage point of another vehicle, or from a point completely independent of any vehicle. Similarly, a maintenance trainer virtual world viewer may be focused on a specific part from a particular angle or a different part or from a different angle. A maintenance trainer may also have a viewer which shows the value of different instruments such as a voltmeter or oscilloscope. Set Perspective allows the ITS to make adjustments to different displays to different views or perspectives. Play causes the recorded scenario to be replayed from the current time faster than, slower than, or the same as real-time. Freeze halts the replay and freezes the displays.

Associated with replay may be the notion of logged annotations. If the simulation supports display of logged annotations during replay, it should also support the ITS making these annotations during the simulated scenario. The ITS issues Log Annotation <scenario time> <URL> service requests which contain a time stamp and URL of html pages that describe an event or instructionally relevant information for the debrief.

The optional capability (LSUI) to facilitate dialog between the ITS and student using the simulation's user interface elements is provided by a pair of additional service requests, LSUI Feedback and LSUI Interactive Feedback both sent via Action Request message similarly to Feedback and Interactive Feedback service requests. The format is LSUI Feedback <URL> <element> <mode> where the simulation upon receipt will display the html page at the URL and place the user interface element into the specified mode. The elements are defined in documentation provide by the simulation developer. Examples are entities, units, areas, and instruments. The element might include subidentifiers, such as the specific part of a simulated instrument. Mode could represent color, intensity, or highlighting, such as flashing. In the case where element was an audio channel the mode could be a tone or even a .wav file name to play. LSUI Interactive Feedback <URL> <list of (<element> <identifier> <mode>)> causes the simulation to display the page at the URL, and place each listed element in its corresponding mode. This can be used to highlight for the user what his options for selection are. If the user selects one of the elements, the corresponding identifier is returned in an Action Response message. If the selection is cancelled, then that fact is returned instead of a selection.

The optional capability (LCSE) to integrate the simulation's scenario editor and the ITS's additional scenario information editor would be used to allow coordinated scenario entry and editing by Subject Matter Experts (SMEs). Examples of additional information possibly needed by the ITS from the SME are: parameters that might appear in the student's pre-scenario briefing such as rules of engagement, specific routes, points, or areas, reported symptoms of equipment in need of repair, etc.; information that the ITS needs that the student shouldn't see such as key terrain, correct and common incorrect solutions such as plans or diagnostic steps, and annotations of those solutions; and descriptions relating to the applicability of the scenario for different types of students such as which principles and skills it tests/practices and how hard it is.

For LCSE, both the simulation scenario and the ITS additional information editors must be able to be invoked from the command line with an optional parameter, the file name where the simulation's scenario is stored. The ITS scenario editor must be able to map from the simulation's file name to its own files for the same scenario.

Data that must be transmitted between the editors occurs as described above for simulation data (XML inside of either DIS Experimental PDUs or an HLA RPR FOM extension object). This includes notification from the simulation's scenario editor of when the scenario file changes (in either its name or contents when it is saved to disk) or when the scenario changes in memory. These changes include adding, deleting, or changing elements such as military units, briefing parameters, plans and orders, graphics, symptoms, etc. The two scenario editors will each have their own user interface in their own separate windows. However this communication from the

simulation's scenario editor will allow the ITS's editor to prompt for additional information as elements are added to or changed in the scenario.
The ITS may need information from the SME most conveniently added using the simulation's tools provided to students. For example, the simulation may provide an interface for the student to enter plans (or repair actions). If the ITS needs the SME to enter correct and common incorrect plans (or repair actions), this should be done from the same interface. The required data will already be available from the Level 2 interface. The ITS editor simply needs to prompt the SME and get some additional information from the SME (such as whether this is a good or bad plan (or action sequence)) about the information being entered. Similarly, if the optional LSUI capability has been implemented, the ITS editor can use it, in addition to the ITS using it during training, to prompt the SME by highlighting elements of the plan and asking for additional annotations such as the related principles and the rationale as to why the element is good or bad. It could ask the SME questions, such as which enemy unit should be considered most dangerous, and receive answers using the simulation's interface, such as clicking on the enemy unit in the PVD. Similarly, this sort of interaction may also be useful with the simulation's scenario editor. For this purpose two additional service requests, LCSE Feedback and LCSE Interactive Feedback, which are exactly analogous to LSUI Feedback and LSUI Interactive Feedback are defined.

Additional items can be defined by the simulation developers. If they want to make additional data available from the simulation, this can just be defined in the XML and sent out in DIS Experiment PDUs or HLA Simulation Data Interactions as described above. If the simulation or ITS developers want to make additional services available, they just become additional service request types passed in Action Request message also described above. An example would be if the simulation actually did some evaluation of the student's actions on its own. In that case, this would be additional data from the simulation that the simulation developers could define.

Part of the I/SIS standard is required documentation. Since some amount of freedom is allowed for in the exact format and content of some of the XML-represented information, each type of data and service request has to be documented with multiple realistic examples of each provided.

I/SIS Summary

The above described prototype standard is summarized in the following five lists. (Short definitions of the levels and use cases are also repeated for convenience). To be considered compliant, all systems must include the All Use Cases list. Each system must also include the lists for their applicable use cases. Each list is divided into the applicable levels. Level 1 applications must only support the level 1 portion of the list but Level 2 applications must support both Level 1 and Level 2 portions.

Level 1 – Basic Integration
Level 2 – Advanced Integration
LIDR – ITS Driven Replay
LCSE – Coordinated Scenario Entry
LSUI – ITS partial control of Simulation User Interface
TDM – Tactical Decision Making use case
EOM – Equipment Operations and Maintenance use case
IC – ITS Centered use case
SC – Simulation Centered use case

All Use Cases:
- Level 1
  - Service Requests (SR) via Action Request messages
  - Feedback SR
  - Developer Created Documentation of Interface
- Level 2
  - Interactive Feedback SR
  - Controlling component sends and other accepts Start/Resume & Stop/Freeze SIMAN messages
  - UUID Student IDs
  - Logon SR from controlling component
  - Log Annotation SR
- LIDR
  - Set Time SR
  - Set Perspective SR
  - Play SR
  - Freeze SR
- LCSE
  - Command Line Start of Sim & ITS Scenario Editors
  - Sim notifies ITS of scenario changes
  - Level 2 implemented
  - LSUI implemented
  - LCSE Feedback SR
  - LCSE Interactive Feedback SR
- LSUI
  - LSUI Feedback SR
  - LSUI Interactive Feedback SR
  - Additional Items
  - XML Data and SRs as required

TDM:
- Level 1
  - DIS or HLA RPR FOM
  - ITS access to additional scenario-related ITS information
- Level 2
  - XML Data in Experimental PDUs or HLA Simulation Data Interaction in I/SIS FOM
  - Orders in XBML, Audio in files/XML, other communications/actions/context in XML

- MSDL & XML Scenario Files

EOM:
- Level 1
  - XML Data in Experimental PDUs or HLA Simulation Data Interaction in I/SIS FOM
  - XML formatted lists of control actions and instrument values
- Level 2
  - XML Scenario Files
  - ITS access to additional scenario-related ITS information

IC:
- Level 1
  - Command Line Sim Start (scenario file)
- Level 2
  - ITS sends and Sim accepts Reset, Load Scenario, & Start AAR SRs
  - Entity control via HLA Ownership Switch or DIS Set Data

SC:
- Level 1
  - Command Line ITS Start (scenario file)
- Level 2
  - Sim sends and ITS accepts Evaluation, Coaching, and Debriefing SRs,
  - Sim Sends and ITS accepts Assign Team Member SR

6. Examples of Applying the Standard

All three examples mentioned above, BC2010 ITS, IFT, and FCS ITS are past systems which have already had their simulations integrated with their ITSs, obviously, before the creation of I/SIS. In fact, each of those systems partly contributed to the requirements listed in Section 4. In this section, the integration that would have been performed if I/SIS had already existed will be described.

Taking the BC2010 ITS system as an example, the developers would first consult the I/SIS Summary lists. Specifically, as a TDM SC system, they would need to include items from 3 lists - All Use Cases, TDM, and SC. Considering first a Level 1 integration, the simulation developer would have to support either DIS or HLA and the RPR FOM. BC2010 already supported the latter. The only other L1 requirements on the simulation would be to be able to accept Feedback service requests via HLA Action Request messages. BC2010 would have to be able to display the html pages referenced in those requests. BC2010 would also need to invoke the ITS via a command line and pass the scenario file name.

For Level 1, the ITS would have to be developed such that it could be invoked via a command line where the first argument is the simulation's scenario file name. Using that file name, the ITS would need to open its own files associated with that scenario. In this case, this would be descriptions of key areas for that scenario's terrain and tactical situation. This included avenues of approach and logical blocking positions. It would evaluate student performance based on the data available in the RPR FOM (mostly unit movements in this application). The ITS would assemble mission execution real-time and debriefing feedback as html pages and send them to the simulation via Feedback service requests in HLA Action Request messages. Although always required to some degree, there would be no significant documentation for this integration from either developer.

Unfortunately, Level 1 integration will not achieve the capabilities realized by the actual application, since it also debriefed the student's plan, so a level 2 integration is called for. Again, consulting the same 3 lists, for Level 2 this time, the simulation developer needs to output orders (the student's pre-mission plans and real-time orders) in XBML format inside an HLA Simulation Data object defined in the I/SIS FOM extension to the RPR FOM. At this point greater integration will have already been achieved, since the BC2010 can start the ITS application on its own and the simulation can display the html feedback to the student from within its own interface.

Correspondingly, the ITS developer would need to accept those XBML formatted orders. This would probably be easier than what was actually done, which was to receive the orders in the proprietary format of the simulation developer, requiring the simulation developer to provide decoding software. To fully support Level 2, the ITS should also accept Start/Resume & Stop/Freeze SIMAN messages.

Neither application had a logon process so that service request is irrelevant. BC2010 did include a scenario replay option so it should support the Log Annotation service request. Based on previous messages from BC2010, the ITS would create html descriptions of instructionally relevant events (such as mistakes) and log them for display during the playback. The ITS must support requests from BC2010 to turn on or off specific evaluations, coaching, and debriefing. This would allow the student or instructor to choose different modes of training through BC 2010. For example, a novice might need everything turned on, but a more experienced student might want the more realistic practice afforded by having coaching turned off. He might still want debriefing in order to catch any mistakes he might make. The level of detail that BC210 provides for the instructor

or student to turn these on and off would be up to the simulation developer's discretion. However, the ITS should provide the finest detail possible and document the options available through these service requests.

For a single user of BC2010, the main context is what units are displayed in the Plan View Display (PVD). At the beginning of the scenario run, BC2010 creates an XML list of all units displayed in the PVD, places it into an HLA I/SIS FOM extension Simulation Data object. Each time a change occurs, because units appear or disappear, move into/out of the map's display, or the map is panned or zoomed in/out by the student, the changes are sent in an updated Simulation Data object. The ITS accepts this information and can, at its discretion use it as additional information when evaluating student actions. For example, the ITS may determine that the student's failure to respond to an enemy's movement was caused by the fact that the PVD does not currently show the relevant enemy units.

The Level 2 BC2010 should accept Interactive Feedback service requests. The ITS could, at its discretion, choose to use them or not. For example during debrief, instead of just pointing out an error, such as failure to adequately allocate additional units to a blocking position in the face of a larger than expected enemy attack, the ITS could ask some questions first, such as did the student notice that the attack was larger than expected. The student's yes/no answer would be extracted by BC2010 and sent back in XML via an Action Response message. A "no" would lead to different remediation dialog than a "yes". For example a "yes" would lead to questions about whether the student considered the relative force ratios. A "yes" to that would quiz the students about which units he considered to use as reinforcements and why he rejected each. Ultimately he would have been instructed that one of the units he rejected had an irrelevant or lower priority mission than the blocking mission. A "no" to the original question would cause the ITS to provide information about the need to monitor key parts of the battle and to be primed for differences from what was expected.

The above discussion naturally leads to the desire for the optional LSUI capability to allow the student to respond to some of the ITS's questions by clicking on elements in BC2010's PVD. For example, the ITS might ask the student which enemy unit he perceived as most dangerous by issuing the service request, LSUI Interactive Feedback <URL of html page asking "which enemy unit is most dangerous? Select one from the PVD."> <list of all enemy units(<enemy unit i> <enemy unit i ID> <highlight mode>)>. BC2010 highlights those enemy units and the student selects one whose ID is passed back to the ITS in an Action Response message.

Level 2 dictates that BC2010 scenarios are stored in MSDL (an XML) format. The ITS could theoretically read and write them. Similarly the ITS scenario information would be stored in XML format. For the BC2010 ITS this is the description of key areas for specific scenarios. The BC2010 developers could easily allow scenario authors authoring scenarios in the BC2010 scenario editing environment to also specify these areas on the scenario map and store them into the ITS's scenario information format, to realize a single editing capability for both parts of the scenario. Optionally LSCE could provide for integration of the separate editing software packages, but this would not have a big advantage for this application.

The IFT, as an EOM IC system must address the All Use Cases, EOM, and SC items. Considering first a Level 1 integration, the flight simulator developer will have to provide XML formatted lists of control actions and instrument values via either DIS or HLA. For this example assume that the developer chooses HLA so that the action and instrument value lists will be transmitted in an HLA Simulation Data object. The action list consists mainly of the current cyclic and collective positions plus several others of lesser importance. The instrument list will include the following list of instruments and their values: Altitude indicator, Airspeed indicator, Climb rate, Turn rate, Heading, Attitude indicator - pitch, Attitude indicator – roll, etc. The specific name of the instruments and controls and the specific meaning of the values must be documented by the developer with multiple examples of each. The flight simulator must be able to be invoked from the command line and accept the scenario file name to load as its first argument. Finally the flight simulator must accept the Feedback <feedback URL> service request via an HLA Action Request and display the feedback html.

A student using the IFT logs on through the ITS portion. The flight simulator, unlike in the current version of IFT, is automatically started for him in the ITS chosen scenario. Real-time feedback and after action debriefing is presented through a window provided by the flight simulator. While this may be an acceptable situation, audio feedback is more appropriate for this application. Additionally the current IFT can flash instruments as a hinting mechanism. These two capabilities would both be handled by the optional LSUI service request, LSUI Feedback <URL> <element> <mode>. To provide real-time audio coaching the element would be "Audio" , the URL would be empty, and the mode would either be a quickly generated .wav file or a string to be spoken by a speech generation component in the flight simulator. The developer of the flight simulator could decide which. To flash the Climb Rate instrument the ITS would send, via an Action Request Message, LSUI Feedback < >

<Climb Rate> <Flashing>. The modes would be clearly documented for each instrument by the flight simulator developer.

Additional useful capabilities would be provided with a Level 2 integration. The flight simulator has no logon or password procedures so those elements of Level 2 are irrelevant. The flight simulator would have to support Start/Resume and Stop/Freeze SIMAN messages. This would be useful when the ITS discerned the need to explain something to the pilot in detail in the middle of a scenario run.

For Level 2, the Flight Simulator should also provide documented facilities to control the simulated helicopter. If the student lost control of the helicopter, the ITS would issue the HLA request to take ownership of the Cyclic and Collective attributes then set those controls to a series of appropriate values to get the helicopter back to a state that the student could handle.

If there was a desire to include replay in the debrief, then the LIDR integration option could be selected. The flight simulator developer would have to create a replay capability, accept the Level 2 service request, Start AAR, and accept the LIDR service requests. During the scenario the ITS would note interesting time periods to replay to the student. After the flight exercise was over the ITS would issue a Stop/Freeze and a Start AAR service request to inform the flight simulator that the exercise was over, control inputs should be ignored, and a debriefing, possibly to include replay would start. For each time period the ITS would issue a Set Time, Play, and Freeze service request. Set Perspective would be used, if allowed by the simulator, to set the perspective outside of the helicopter, if that served to illustrate some point better.

7. References

[1] Stottler, R. and Spaulding, B., *Requirements of an ITS/Simulation Interoperability Standard (I/SIS)*, Fall SIW, Sep. 2004

[2] Jensen, R., H. Marshall, J. Stahl, R. Stottler (2003) "An Intelligent Tutoring System (ITS) for Future Combat Systems (FCS) Robotic Vehicle Command", I/ITSEC 2003.

[3] NTMF Website, http://www.ntmf.com/

[4] Layman, G. et. al., *C4I-Simulation Interoperability Using the DII COE and HLA*, 6th ICCRTS, June 2001, http://www.dodccrp.org/events/2001/6th_ICCRTS/Cd/Tracks/Papers/Track3/113_tr3.pdf

[5] Roberts, J. and Dobbs, V., *Application of a C4I Reference or Starter FOM to an Existing Simulation Environment*, Fall SIW, Sep. 2000

[6] SISO Coalition Battle Management Language Website,http://www.sisostds.org/doclib/doclib.cfm?SISO_RID_1005560

[7] Sudnikovich, W., Pullen, M., Kleiner, M., Carey, S., *Extensible Battle Management Language as a Transformation Enabler*, George Mason University, November 2004

Author Biographies

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 the 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 US Marine Corps Combined Arms Command and Control Training Upgrade System (CACCTUS). He has a Masters degree in Computer Science from Stanford University.

BRIAN SPAULDING is currently the Director of Contract Engineering for MÄK Technologies, Inc., managing the performance and technical aspects of all engineering services contracts. Mr. Spaulding's projects include the DARPA-sponsored DARWARS program, the SIMinterNET family of Intermediate Desktop Simulations, the development of 2D and 3D visualization enhancements for CECOM and AFRL, and the development of an after-action review system for the UK Apache Program. He has worked at GTE Government Systems as the lead architectural engineer responsible for the development of a SGI-based, Collaborative 3D virtual environment and supporting the development of GIS and digital mapping tools.

ROBERT RICHARDS, PH.D. is a project manager at Stottler Henke Associates, Inc. He has been and is the principal investigator on projects dealing with training, simulations and integration for the Navy, Air Force and NASA. Currently he is working on a PC-based training simulator teaching crewmembers (pilot, co-pilot and sensor operator) the new MH-60S and MH-60R platforms. The training simulator can be used solo, however, it is designed to be integrated with other hardware/software including Microsoft Flight Simulator, COTS stick, throttle and rudder pedals, as well as head tracking products and COTS head-mounted displays. Dr. Richards received his Ph.D. from Stanford University.