Integrating Simulations into Sharable Content Object Reference Model Learning Environments

Curtis Conkey	Brent Smith	Chris DuBuc	Peter A. Smith
NAVAIR / NETC	ECS	ECS	JHT
Orlando, FL	Orlando, FL	Orlando, FL	Orlando, FL
curtis.conkey@navy.mil	brents@ecsorl.com	chrisd@ecsorl.com	peter@smithpa.com

ABSTRACT

The United States Navy is engaged in an enterprise-wide transformation of how it trains. One key component of this transformation is the development of the Navy's Integrated Learning Environment (ILE). This initiative uses a variety of instructional development strategies to meet the diverse requirements of the Navy's workforce and assures content which is relevant, current, accurate and engaging. Where content design and development are concerned, the Navy has mandated the use of the Sharable Content Object Reference Model (SCORM) guidelines for all learning materials to be used within the ILE.

However, the growth in capability and flexibility of interactive simulations has led to a challenge for the Navy, regarding how to integrate the power of these interactive simulations into the ILE. The Naval Education and Training Command's (NETC) Experimentation Lab, located at the NAVAIR Training Systems Division in Orlando Florida, is experimenting with the tracking of learning objectives between a Learning Management System and a gaming/simulation engine for reporting back progress towards meeting defined training objectives. Additionally, efforts are looking at the challenge of objectively tracking training objectives in three dimensional, interactive simulations and/or serious games. In order to meet these requirements, technology needs to be developed to allow in-depth assessments of student performance against established training objectives while the simulation is running.

This paper discusses in detail the underlying technology being developed to launch, track and manage training simulations using the Delta3D game engine. It identifies the current state of technology in supporting this role, as well as supporting technologies and foreseeable challenges. Finally, it provides a detailed report of the lessons learned from this effort.

ABOUT THE AUTHORS

Mr. Curtis Conkey, from NAVAIR, Orlando, is the Lead Engineer for the Naval Education and Training Command's (NETC) Experimentation Lab. As part of NETC's N9 Learning Strategies Division, the lab's primary charter is to experiment with emerging technologies that have application to the US Navy's training requirements. A recent focus has been on Open Source Solutions. The NETC lab is also involved with the experimentation in the serious games arena and with standards bodies researching next generation interfaces for interactive simulation. Mr. Conkey has presented at various conferences on the lab's efforts. He also has prior experience in conventional simulations systems having worked with *"On-Board Team Training Applications"* for the Next Generation Virginia class submarine. Prior to his Navy career, he completed 20 years with AT&T / Lucent Bell Laboratories where he managed the creation of next generation networking solutions for high profile customers. Mr. Conkey has a Bachelors Degree in Electronics Engineering, a Masters Degree in Computer Science and is currently a Doctoral Student in Modeling and Simulation at the University of Central Florida. He is a member of the Association of Computing Machines (ACM) Special Interest Group for Simulation (SIGSIM), Society for Modeling and Simulation (SMS) and the Simulation Interoperability Standards Organization (SISO).

Mr. Brent Smith is the Vice President and Chief Technology Officer for Engineering & Computer Simulations, Inc. While at ECS, he has performed extensive research in the areas of collaborative distributed learning architectures, distributed simulations, and the use of commercial gaming technologies as educational tools for the US military. His professional focus is on the research and development of technologies to enable new learning methods and tools to improve training effectiveness. Mr. Smith is a frequent speaker on the topic of advanced learning technologies and has spoken at many conferences even testifying before Congress. He has also published numerous articles and papers on the topic as well. He is a member of the Association of the US Army (AUSA), Military Operations Research Society(MORS), the National Defense Industry Association (NDIA), and the Interservice / Industry Training, Simulation and Educations Conference (I/ITSEC) Subcommittee for Education. Currently, he is also secretary and technical writer for a collaborative effort between IEEE Learning Technologies Standards Committee and the Simulation Interoperability Standards Organization which was formed to identify potential standards for the key interface points between simulations and SCORM managed learning environments.

Mr. Chris DuBuc is a senior software engineer and project manager for Advanced Learning Technologies with Engineering & Computer Simulations. He is currently directing work efforts that involve using the Delta3D game engine within the Navy's Integrated Learning Environment. He has extensive knowledge of the evolving standards and operational requirements involved with using games and simulations within a managed learning environment. Prior to joining ECS in 2005, Mr. DuBuc ran his own consulting company, Buke Inc., where he created custom, web-based software solutions for both PC environments and Pocket PC environments specifically focusing on web-services and database connectivity. Mr. DuBuc was also V.P of Florida operations for Sage Software, where he oversaw development of a wide range of web-based software solutions for the telecommunications and mining/exploration industries. Mr. DuBuc has a BS in Management systems with a minor in Industrial Psychology from Rensselaer Polytechnic Institute (RPI).

Mr. Peter Smith is a games and simulation software engineer for JHT incorporated, contracted to work in the NETC Experimentation Lab, located at NAVAIR Orlando. He concentrates primarily on Serious Games and other emerging technologies and how they will affect the future of Naval training. He is also a volunteer with the Serious Games Initiative where he acts as a community manager for both serious games and games for health communities. Mr. Smith graduated from Rose-Hulman Institute of Technology in 2002 with a BS in Computer Engineering and a Minor in Computer Science, with a specialty in Computer Imaging Systems. He graduated from the University of Central Florida in 2005 with a Masters in Modeling and Simulation and is currently pursuing his PhD. in the same field. Mr. Smith has written a multitude of book chapters for various publishers on topics such as Serious Games, Massively Multiplayer Online Games (MMOGs), and the history of games and game-based training.

Integrating Simulations into Sharable Content Object Reference Model Learning Environments

Curtis Conkey
NAVAIR / NETC
Orlando, FL
curtis.conkey@navy.mil

Brent Smith
ECS
Orlando, FL
brents@ecsorl.com

Chris DuBuc
ECS
Orlando, FL
chrisd@ecsorl.com

Peter A. Smith
JHT
Orlando, FL
peter@smithpa.com

INTRODUCTION

The United States Navy is engaged in an enterprise-wide transformation of how it trains. One key component of this transformation is the development of the Navy's Integrated Learning Environment (ILE). In December of 2002, the Naval Education and Training Command (NETC) established the ILE to help launch, track and manage almost 4000 e-learning courses for approximately 1.2 million active-duty Sailors, Marines, Department of the Navy civilians, Reservists, retirees and family members enrolled in the Defense Enrollment Eligibility Reporting System (DEERS)(Maskell, 2003). This initiative uses a variety of instructional development strategies to meet the diverse requirements of the Navy's workforce and assures content which is relevant, current, accurate and engaging. It combines support tools for developing and distributing electronic course materials, and managing student and curriculum records, with standards for classifying content, formatting files, and interoperability among other systems.

Another component of this transformation is the use of gaming technology and simulations to help improve the effectiveness of Navy training. Research shows that people perform better after instruction when they have learned in the context of doing (Kelly, 2002). The Open Source Delta3D project is a gaming and simulation engine developed through the Modeling, Virtual Environments, and Simulation (MOVES) Institute at the Naval Postgraduate School. This engine, and other's like it, has great potential to support adaptive learning by placing learners in a "real-world" environment and allowing them to learn in context. While these types of simulations offer the opportunity to immerse a student within a context-based scenario, they do not, by themselves, constitute training. Game-based learning applications and simulations are only part of the mix of learning strategies that must be experienced by the learner in order to create a well trained student. Student tracking and assessment are important elements in the learning process and play a key role in determining when the student is ready to proceed to a more challenging level of training. This also has the potential to free-up instructor resources by allowing more automated training and accessments of student skills thus allowing instructors to focus their attention were needed.

In order for game-based training applications and simulations to meet these requirements, technology needs to be developed to allow in-depth assessments of student performance against established doctrine in order to evaluate student proficiency and provide feedback to both students and instructors. The NETC Experimentation Lab in Orlando, FL is experimenting with methods to determine how to best integrate these game-based training applications and simulations into the Navy ILE. A key element of this effort is the interface between a Learning Management System (LMS) and a gaming/simulation engine for reporting student progress towards meeting defined training objectives. This paper will discuss the underlying technology of the SCORM-SIM research project that is allowing an LMS to launch, track and manage training simulations using the Delta3D engine. It will identify the current state of technology in supporting this role, as well as supporting technologies and foreseeable challenges in achieving the goals of this project.

OVERVIEW OF THE NAVY'S INTEGRATED LEARNING ENVIRONMENT

From an Information Technology perspective, the Integrated Learning Environment uses a Service-Oriented Architecture (SOA) to decentralize key system functions. The ILE currently consists of the following applications:

1. Navy Knowledge Online Portal – Acts as the single point of entry to the ILE
2. Learning Management System (LMS) – Uses the ThinQ Training Server LMS and contains the ILE master catalog for accessing education and training content
3. Learning Content Management System (LCMS) – Uses OutStart Evolution Learning Manager and provides for creation, storage, reuse, management and delivery of learning content

4. Corporate Enterprise Training Activity Resource System (CeTARS) – Consolidation of independent schoolhouse and training information systems
5. Navy Training Management and Planning System – Data warehouse for all training and education information for a user
6. Skill Object Data Mart – Repository of required skills, abilities, tasks and knowledge required to perform Navy jobs and duties
7. 5 Vector Model (5VM) – Defines parameters around which a Sailor's personal and professional development is designed

The ILE integrates applications developed by the training and education commands. These applications operate within the layered architecture of the Navy's information infrastructure and must be built according to common interface specifications (NPDC, 2004). The Navy Knowledge Online portal provides the primary interface for students to access such applications as needed.

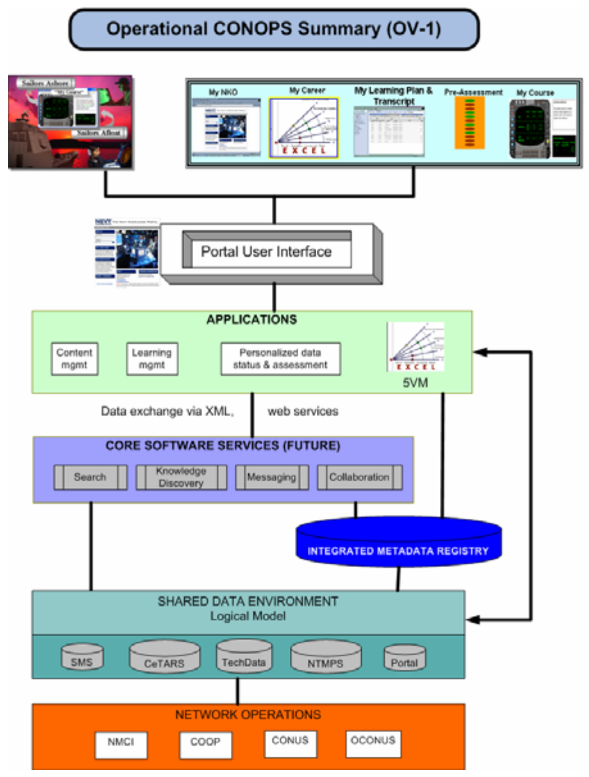

Figure 1: ILE-ISA layered architecture of the Navy training and education planned information infrastructure (NavyILE(1), 2006)

Where content design and development is concerned, the Navy has mandated the use of Shareable Content Object Reference Model (SCORM) guidelines for learning materials to be used within the ILE. Critical to content is the need to have processes in place to ensure updates are quickly realized across the entire system. SCORM 2004 promotes reusable and interoperable learning resources, while also providing the ability to define complex instructional logic across Sharable Content Objects (SCOs). Within the SCORM context, the ILE is a server-based environment in which the intelligence resides for controlling the delivery of learning content to students.

STANDARDS FOR ILE CONTENT

The SCORM Content Aggregation Model (CAM) relates specifically to the assembling, labeling and packaging of learning content. The CAM is based on the concept of a "Content Package", which is a self-describing archive that bundles related learning content with one or more XML-based manifest files. A Shareable Content Object (SCO) is the basic building block of the CAM. A SCO is a collection of assets developed to provide the instructional requirements of a Learning Objective. The Navy has mapped a SCO to Enabling Learning Objectives (ELO) and in its absence, Terminal Learning Objectives (TLO) (Navy ILE, 2004). A Terminal Learning Objective can be defined as an expectation of what a student will know at the end of a lesson or unit. They typically consist of tasks, conditions and standards, whereas Enabling Learning Objective are more focused on the major task steps that comprise the TLO.

The SCORM Run-Time Environment (RTE) specification handles the launch of learning content, communication between content and an LMS, data transfer and error handling. SCORM currently specifies that an LMS must launch learning content as an HTML web page on the client. At this point, the LMS is also responsible for establishing a "handshake" with the client browser and for providing it with access to an Application Program Interface (API) during each learning session. This API represents a set of predefined functions, which the browser can use to modify values on the LMS server. The SCORM RTE also defines a specific data model, which is the required vocabulary that must be used when communicating with the LMS via the API.

Finally, the SCORM Sequencing & Navigation specification describes how an LMS manages the delivery order of learning content. A basic branching model is supported that allows for content redirection based on student input and performance.

A VISION OF NEW CAPABILITIES AND TECHNOLOGIES FOR THE ILE

Recent years have seen huge increases both in computing power and the number of people able to access computers and the Internet. This proliferation of information and communication technologies has enabled higher quality learning to be made available to an ever-growing audience through increasingly sophisticated modes of presentation. The increased use of simulations across the services supports the idea that training through practice in realistic context-based scenarios help influence skilled decision-making. Traditional or conventional training programs use a variety of instructional development strategies to support a student's need to master a variety of competencies. Game-based training applications and simulations have great potential to become powerful and valuable extensions to traditional educational initiatives. Cognitive skills can be practiced and honed as students interact within virtual environments.

The ILE strategy establishes a seamless training management and delivery capability in support of providing learner-centric training to the right people and in an appropriate manner at the point of need. Current legacy training management systems and new emerging technology systems must be fully and seamlessly integrated to provide a single access and management point for all events and activities (Navy ILE (2), 2006).

Assessment strategies should, in the future, include opportunities to continually monitor progress against a discrete set of metrics. They should include the ability to modify and insert new resources into a learning strategy as required in order for learners to learn from their performance. As a student's level of proficiency increases, the learning strategy should conform to the evolving skill level of the student and different feedback formats should provide immediate knowledge of performance thereby increasing the rate of acquisition and retention of learned behaviors.

Attention must also be paid to new collaborative technologies, such as lobby services to allow interactive training for interoperable and dispersed teams. These collective applications will need to have the ability to track and assess individuals and teams at varying levels as well. Skillful management of new information technology must play a role in the solution. These technologies should be transparent to the user, while still providing interactive experiences that transfer efficiently into high levels of performance.

HOW WILL GAME TECHNOLOGY AND SIMULATIONS INTEGRATE INTO THIS MIX?

The Navy's "Revolution in Training" uses performance-based training to give Sailors the knowledge, skills and abilities needed to accomplish their mission. As part of this effort, the Navy is considering deployment of PC-based simulations to support training transformation with the overall goal of improving performance while reducing costs. There have been many changes in the last decade in the use of simulations and computer games for training and education. The recent advances in PC processing power, graphics cards and the accessibility of broadband internet connections increase the likelihood of using these types of applications for training.

In the past, there has not been a consistent approach to how these game-based training applications and simulations are developed; none of the major courseware initiatives, such as re-use, interoperability and sharing have been addressed (Navy ILE (3), 2006). Any PC simulations developed for training via the Internet will likely be accessed on the Navy Knowledge Online portal and integrated with the LMS and LCMS. SCORM conformance will also be a requirement. Bandwidth limitations may limit the level of fidelity in these products, but should not limit the level of interactivity afforded (Navy ILE (3), 2006). Due to the complexities of deploying and installing a simulation application onto a computer from the Internet, it is anticipated that the delivery of these applications will need to be managed by a self-contained, browser based application.

Recognizing the importance of these requirements, two IEEE standards committees (The IEEE Learning Technology Standards Committee (LTSC) and the Simulation Interoperability Standards Organization (SISO)) have formed a collaborative study group to investigate the potential of formalizing a standard set of technical specifications to allow simulations and/or games to be launched and managed through SCORM conformant learning environment.

As this technology becomes more advanced and costs are reduced, we theorize that the training community will embrace games and simulations as another media type that may be used to convey instruction. The project discussed through the rest of this paper will take a look at the first steps being taking to make this possibility a reality through the development of a prototype application using similar Open Source technologies.

OVERVIEW OF THE SCORM-SIM INTEGRATION PROJECT

Given the perceived benefits from using games and simulations within the Navy's ILE, a study was funded to investigate and validate technologies that could be used for integrating them into the ILE. This project is addressing many issues, both pedagogical and technical. From a technical perspective, it is attempting to identify the key interface points between simulation content and LMS environments, such as delivering simulation content to the learner, monitoring key interactions and performance within the simulation content and determining what the student should next experience within the continuum of training.

In the early stages, this project is leveraging ongoing development programs with the Open Source Delta3D engine. As an open source effort, it is free to use; it has neither licensing nor distribution costs. It is however, a fully featured 3D development environment comprised of multiple "best-of-breed" open source projects. One major goal of the Delta3D project is to provide a low cost platform for the development of military training applications.

One such project being leveraged for the SCORM-SIM Integration effort is the development of Delta3D After Action Review (AAR) capabilities for the US Joint Forces Command. This project plays a vital role in the simulation pipeline by supporting multiple levels of mission debriefing and analysis for a particular training event. There are two main components of an AAR system which are being addressed in this JFCOM project; logging (event recording and playback) and objective tracking (task and subtask completion). The SCORM-SIM Integration project will leverage both components of the AAR project to accomplish the objectives of the SCORM-SIM Integration project.

INITIAL TECHNOLOGY BARRIERS

The benefits of a web-based distribution model are well known. However, these same attributes often serve as impediments to the creation of more highly interactive training experiences. For example, storing content on a centralized server requires that it be divided into small enough components to be delivered on-demand over the network. While this may not be an issue for simple web pages, it proves to be quite an obstacle when dealing with resource-hungry training scenarios like simulations and game based training. While web browsers and Internet technologies make application deployment almost trivial, this same technology strictly limits an application's access to a client machine's system resources.

Learning Management Systems are based on the concept of a centralized server delivering learning content to users over a network, thus enabling learning content to be widely distributed to anyone with a web browser and access to the system. The training material itself generally consists of collections of web pages, possibly interspersed with video, sound, and/or Flash-based animations. Within the SCORM context, this content can be described, sequenced, tracked and delivered in a standardized fashion. The requirements of doing this with simulation will vary, and different approaches and models for use of simulation may result in different solutions (CMU).

Regardless, Game-based training applications and Simulations will often have more complex requirements for data tracking and state management than what is currently possible within SCORM. In order for student performance to be tracked inside a simulation, the data communicated to the LMS must be confined to the assessment data model that SCORM provides. A mechanism for assessment needs to be developed to allow student performance data to be extracted from a simulation and communicated to a LMS. Generally speaking, each SCO can contain a number of "objectives", and learner progress toward each objective can be tracked according to basic data values such as completion status, success status, and score. While a simulation scenario may track many assessment variables internally, it needs to be able to combine these variables into data values that an LMS is able to understand.

An LMS is limited by SCORM in how it is able to interact with a SCO, and consequently a simulation. The LMS is not able to "watch" what is happening inside a SCO and take an action. When the focus is shifted to interactive simulations, these issues are amplified. Simulations often depend upon the processing power of the local client machine. A 3D simulation requires direct access to a machine's graphics subsystem in order to render an environment in real-time, and to the local file store (i.e. hard drive) in order to store and quickly access needed resources. Web browsers and SCORM are not designed to handle this level of "system awareness."

Adding to the difficulty is the fact that often a simulation's resources necessitate a large installation footprint (possibly 100's of MB) which needs to be configured and available for a simulation-based application to launch. With current technology, it is impractical to download this amount of data each time a user wants to run the simulation; and until recently there have been few options to download, uncompress, install, cache, run, and update large applications from within the context of a web browser.

ARCHITECTURAL CONSIDERATIONS

As the SCORM-SIM project is being developed on top of an existing Delta3D project, the requirements of Delta3D have dictated a number of architectural considerations that needed to be taken into account during the design process. The following is a list of some of the major requirements and how they influenced design decisions:

1. **Open Source** – Delta3D is open source. Therefore any third party technology used for the project must be freely distributable. This precluded us from using any licensed technologies such as InstallShield, which is a commercially available installer product that has a web-deployment mechanism.
2. **Cross Platform** – Delta3D is able to run on both Windows and Linux. Technologies developed for the SCORM-SIM project must also be available for both platforms. This influenced the team to focus on Java for creating the web-deployment functionality, versus using Microsoft solutions, which in most cases only run on Windows.
3. **SCORM Compatibility** – Project requirements specify that the solution adhere to the principles and specifications defined by SCORM to the degree possible without restricting our ability to move forward with this integration.
4. **Web Deployment** – The project required that the simulation be deployed and maintained through a web interface. This technology should be transparent to the end-user. The user would not have to “pre-install” any components from a CD; any updates (code or assets) will automatically be applied each time the simulation runs.

DELIVERY AND DEPLOYMENT

As mentioned earlier, simulations typically require relatively large installation footprints and access to system resources on the local machine. In this sense, these “thick client” applications are more similar to a standard office application than to a browser-based e-learning system. For this reason, the delivery mechanism for these applications has traditionally been via a distribution CD that users must install on each PC that they wish to train on. This distribution model violates the principles of both SCORM and the Advanced Distributed Learning initiative.

One primary goal of the SCORM-SIM project was to investigate methods to deploy, update, and maintain 3D simulations from a centralized server via a browser-based LMS (such as the ILE). In order to accomplish this, the project team looked at an emerging technology commonly referred to as “smart clients”. Smart clients attempt to bridge the gap between traditional “thick client” applications, such as Microsoft Word, and “thin client” (browser-based) applications such as eBay, Mapquest, or an LMS. Smart client applications are similar to traditional applications in that they are installed locally on a user’s machine and as such, are able to make full use of local machine resources. However, a smart client is also able to take advantage of the benefits of a traditional web-based application; they can be stored on a web server and easily deployed to users’ machines on-demand; they can be updated by automatically downloading new or revised components; and they can communicate seamlessly with external web services.

Two competing technologies that enable the development of smart clients are “Click-Once” from Microsoft and “Java Web Start” from Sun Microsystems. Due to the cross-platform requirements mentioned previously, the project team decided to incorporate the Sun/Java technology. Note that although this distribution technology is Java-based, this does not mean that the simulation itself needs to be written in Java. The SCORM-SIM project, as mentioned before, is based on the Delta3D game engine and is therefore written in C++; however it is compiled into a library (.dll/.so) that it can be launched from within a Java wrapper application.

To the user, the experience is similar to launching a video from a web page. In the demonstration application developed through this effort, an application developed in Delta3D is launched through the open source Moodle LMS. To do this, Moodle delivers a “launch” web page to the user’s browser, which provides a “Start” button for launching the simulation. When the user first launches the simulation, all of the required simulation code libraries, dependencies, and assets are automatically downloaded and installed into a local cache on the user’s machine. The simulation then launches and runs in a separate window, communicating back to the launch page over a TCP/IP socket. On subsequent launches, the simulation launches immediately. If a content developer decides to update the simulation by modifying code or changing assets, they may simply update the files on the LMS server. Java Web Start will automatically compare files on the LMS with files located on the client machine, and then download and install only the modified files on each user's machine without having to reinstall the entire application.

One additional area of concern that is addressed by using Java Web Start's "smart client" technology is security. For various reasons, simulations will often require elevated permissions and cannot run within the standard internet "sandbox"; and by default Java Web Start will not allow these types of applications to be run on the local machine. However, if the simulation application libraries are digitally signed, the Java technology can be set to prompt the user to accept the download and run the simulation. In addition, using these digital certificates, an IT administrator can set up each machine to automatically allow or disallow a given application.

STUDENT TRACKING AND ASSESSMENT

The assessment process developed through this project is broken into logical components. The simulation raises events as the user interacts with the environment or other entities. An assessment module, developed through the AAR project, listens to these events and uses them to track a learner's progress towards completion of defined tasks.

Delta3D After Action Review Technology

The Delta3D-AAR project has implemented a message passing architecture (as shown in figure 2) that is able to send and process messages between objects in the system (BMH, 2005). A message is a single event within the system, plus all the data that defines that event. Events are generated by components or actors and sent to a Game Manager. The Game Manager exposes methods for sending and processing messages. In the case of a training scenario, the Game Manager might forward messages to the assessment module after an important event has occurred. By enforcing communication through a central interface, there is more flexibility to extend the behavior of the message passing architecture without breaking existing code.

The Game Manager is an abstract layer built on top of Delta3D that manages messages between components and actors, networking and the spatial and logical organization of game actors. Additionally, all messages that pass through the Game Manager are logged and saved for eventual playback through the AAR system. The SCORM-SIM Integration project is leveraging many components of the Game Manager in order to track and assess student performance. Once this project is complete, it will enable the Delta3D engine to be

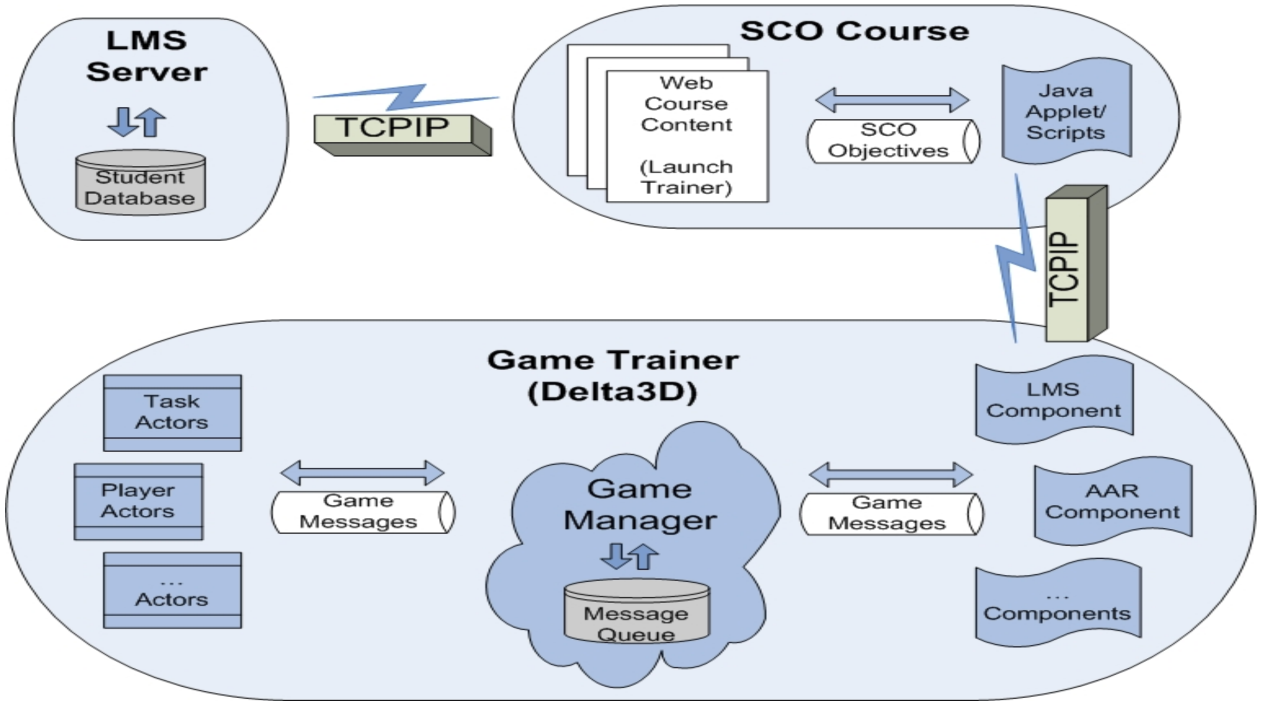

Figure 2: Delta3D Message Passing Architecture – Diagram courtesy of Alion Science & Technology, BMH Operation

deployed, launched and communicate with a SCORM conformant LMS.

Objective Tracking

In order to track a student and assess his or her performance, a simulation needs a basic method of defining a hierarchical set of tasks, conditions and standards for each specific training objective. These are then tied to events within the game or simulation. To accomplish this, the project team developed an assessment model and accompanying code libraries that can be plugged into any Delta3D simulation. These libraries allow a programmer to define and embed assessment “events” into the simulation code; in addition they allow an instructional designer to define a set of objectives or tasks that listen to these events, process them according to a defined set of rules, and forward the results of this assessment on to other systems (such as an LMS).

One objective when designing the architecture for the SCORM-SIM project was to separate the responsibilities of the simulation developers from those of the instructional designer. The simulation developer needs to worry about writing code to make the simulation run smoothly and look good; they should not have to consider every possible scenario that a student may need to be assessed for. Likewise, the instructional designer should not need to worry about changing the core simulation code every time they wish to make changes to the assessment model. The key to maintaining this separation of responsibilities lies in the hierarchical model of assessment developed through this project where events roll up into tasks, which then roll up into objectives.

The Game Manager within Delta3D provides a method that the simulation code can call to raise an event. Events are “things that happen” during a course of a simulation and are determined by the designers to be “relevant” markers of something important; they may encapsulate something as simple as “the user turned left”, or can encapsulate more complex concepts that capture the interaction between simulation objects with other simulation components such as “time” and “state” variables. The complexity of what constitutes a simulation event will ultimately be determined by instructional designers and a simulation’s developers. It is then up to the developer to decide when and how to “code” these events into the simulation so that the assessment module gets notified at the appropriate time. Once these events are flagged within the simulation, they may not be modified without compiling a new version of the simulation code.

Tasks are defined inside the assessment module, which is outside of the simulation code. Each task represents a “unit of work” defined from a list of simulation events and/or other sub-tasks available within a simulation. These events are processed through a rule-based assessment module that has the capability to apply each event to a model that tracks the state of each task (and ultimately each training objective). Because tasks are defined outside of the simulation, they can be created and modified by course designers without having to change and recompile any code.

Example Use Case

More often than not, an instructional designer will need to create complicated tasks that monitor multiple events. These events may need to come in a specific order or within a specific time period in order for the student to successfully complete the task. Consider an example from a medical simulation; a “Patient Triage” task might be defined by events within a simulation that fire when a student cleans a wound, applies a tourniquet, records vital signs, and administers morphine, in the correct order and within 5 minutes.

However, tasks are hierarchical, in that they may contain many levels of sub-tasks. Continuing with the above example, a student might be presented with several patients at once; the instructional designer may wish to define a high-level task that is comprised of not only treating the individual patients correctly but at treating the most critical patients first. In order to assess for successful completion of this high level task, the assessment module will need to keep track of many sub-tasks and the order and timeliness in which those sub-tasks are completed. At this level, a designer may wish to correlate these tasks to overall training objectives as defined through a feature of SCORM called Global Objectives. Global Objectives are a separate structure for learning objectives (i.e., ELOs and TLOs) within SCORM that are able to store the learner’s degree of mastery in the form of a score or a pass/fail state, or they may store the progress of the learner in terms of completion.

Task/Objective Authoring Tools

The importance of the separation between the simulation events and the assessment tasks and training objectives becomes apparent with these high-level tasks. The instructional designer may want to create and modify the tasks over time as assessment needs change. The SCORM-SIM project allows for this via the use of an existing Delta3D level-editor program called STAGE. The STAGE editor allows a user to define the “actors” that exist within a simulation; a specific type of actor for assessment has been developed through the AAR project for use in STAGE, called a “Task Actor”. The STAGE editor can create Task Actors, which essentially provide the capabilities of tasks as defined above. The Task Actors may be linked to events or other sub-tasks, and can be modified and reorganized at will without changing the underlying simulation code.

LMS Component

All of the embedded functionality discussed above would be wasted if it weren’t for the ability of the assessment module to track and store progress information for each student. The requirements of the SCORM-SIM project specified that this data should be able to be stored and tracked in an LMS. To accomplish this, an “LMS Component” was built as an addition to the standard Delta3D library. Similar to the AAR component being developed, the job of this component is to keep track of all the tasks being assessed for in a simulation, and to report a student’s progress on those tasks to an LMS as they are updated.

A designer will typically want the LMS to track and store information on higher-level tasks, because these typically map to the ELOs and TLOs that are already managed by an LMS. To accommodate this, the STAGE editor was modified to allow a designer to “flag” certain tasks as important for LMS tracking purposes. So while the simulation’s AAR component may be tracking many levels of assessment, the LMS Component knows how to filter out and pass on to the LMS only higher-level task updates.

As a final note, the SCORM protocol dictates that the interface to the LMS be embedded in a web page using ECMAScript (JavaScript), meaning that a simulation cannot talk directly to an LMS. Therefore, the project team needed to develop a specialized Java messaging applet along with the requisite protocol so that the LMS Component can talk to the web page that launched it. The launch page (via the Java applet) listens for messages being sent from the LMS Component of the Assessment Module over a TCP/IP socket. When it receives an assessment update message, it translates it as needed to adhere to SCORM protocol and ultimately forwards it on to the LMS. This completes the assessment pipeline: from events triggered in a simulation, to tasks processed within an assessment module, to student progress information being stored in the LMS.

LESSONS LEARNED

Over the course of the development process, a number of lessons were learned; some of the knowledge gained was immediately put to use during the development process, while in other cases it will be used to direct future development efforts.

A significant challenge that the project team encountered was managing the complexity involved with the integration all the various technologies. For starters, to set up a simulation to work per the project specifications required a working knowledge of C++, Java, JavaScript, HTML, SCORM, and the inner workings of the Delta3D game engine was required. All of these technologies must fit together, and there is no single development environment that encompasses everything.

Additionally, several of the technologies depend on setup and/or configuration files. For example:

- Java Web Start requires all files be packaged in digitally-signed JAR files. It also requires that a special application.jnlp setup file be built.
- SCORM requires an imsmanifest.xml file to describe the resources associated with a course.
- The Simulation launch page and its associated Java applet must know which simulation to launch and what tasks it is assessing for.
- Delta3D requires a number of its own and other third-party libraries be installed on the client.

All of these technologies have their own, mostly unrelated settings that must be configured appropriately for the application to work; if something doesn’t work as planned, it is often hard to track down the source of the problem. In order to alleviate some of these integration issues, a “packager” application was developed that sets up a simulation application to work per the project specifications. A content developer can use the packager application to automatically copy and package the required files, write the various configuration files, create the simulation launch page, and so on. Currently, this application relies on a single XML file for input, which makes things much easier than trying to set up the various pieces individually.

Another issue that stems from the nature of deploying a simulation via the web is that of bandwidth. A typical 3D simulation, with its required code libraries and assets, may contain hundreds of megabytes of data. Java Web Start helps by packaging all data into compressed JAR files. While this may reduce the actual footprint, this is still an imposing initial download.

Under current internet speeds, a student may have to wait a substantial period of time the first time they launch a simulation. This issue is not specific to this project and may be addressed in the future with new technologies. In the near term, new policies may be needed to require pre-requisite installations before taking a course. It is not within the scope of this document to discuss new policies. However, the Java Web Start does work when an application is pre-installed via a traditional CD and will still update and modify the simulation components to ensure that the most recent version is the one being run. Additionally, if a course runs multiple simulations that use the same resources, the incremental installs and updates would be negligible and not impact the flow of a course.

Also in the near term, this issue may focus simulation developers on creating more compact and efficient applications, which in itself may be a good thing. In the longer term, the speed of internet access will only increase, and a 50 MB download may be considered commonplace, making bandwidth less and less of an issue for web-deployed 3D simulations.

CONCLUSIONS

One highlight of the development process was the realization that much of the technology needed to accomplish the project goals already existed; most of the development effort went into building interfaces between the various technologies instead of building code libraries from scratch. Additionally, the Delta3D game engine is based on a number of existing open source technologies and Java Web Start has been around for a number of years. One component of this project focused on extending Java Web Start to act as a deployment mechanism for non-Java applications as well.

Likewise, LMSs and SCORM were not developed with simulations in mind, but it was found that they could be used to deploy and manage simulations and track the progress of students who use them. We also should mention some more general technologies that have come into widespread use over the past few years, including XML, AJAX, and Dynamic HTML. The result of all these technologies coming to maturity at the present time allowed us to do things in the SCORM-SIM project that were not previously possible.

The process of researching and developing the SCORM-SIM Integration project has created an enormous opportunity to look at different learning strategies and use cases for using simulations and games within the Navy's Integrated Learning Environment. Most of the technology developed through the SCORM-SIM Integration project was developed such that it can be leveraged and used for other game or simulation engines as well.

In the future, the Navy will have the opportunity to augment "traditional" institutional training by relying heavily on the use of distance learning technologies, fielded simulations and embedded training to meet training and readiness objectives. This fantastic range of possibilities will require far reaching technological innovation to make this a reality.

NEXT STEPS FORWARD

This project is successfully showing the potential of incorporating simulations into a SCORM managed learning environment. However, there is still significant work to be done before this type of integration becomes implemented on a regular basis. The assessment model used for this project was put together to suit the needs of the project. An assessment model based on standards is needed to allow reuse. This formalized assessment model would identify the process for rolling events into tasks and objectives while defining a method for addressing associated parameters, sequencing and dependencies of events within a given task or objective.

One such approach may be to leverage Reusable Competency Definitions (RCDs) to develop this model of assessment. A draft standard for RCDs has been developed to explicitly define and comprehend skills, knowledge, aptitudes, learning objectives and learning outcomes. This pending standard defines a data model that may be used to develop assessment models.

Ideally, these models will be stored dynamically, such that authoring tools can be developed to modify and change them so that the way a student is assessed inside the simulation can be changed without recompiling code. If and when authoring tools are developed, a method of describing events becomes necessary so that instructional designers are able to identify the role of each simulation events. Additional research needs to be performed to determine an appropriate meta-data schema.

Finally, the packager application mentioned earlier was developed to allow easier set up, packaging and configuration of the many components associated with this project. This application has proved to be very useful in this application. However, it is still confusing to create and edit an application for those new to using this tool as it is comprised of a single XML file to enable the required features. One of the future steps envisioned for this project is to provide a GUI to the packager application that will make things more intuitive for those end-users who would actually be deploying these game and simulation applications using the Delta3D engine.

This project has helped highlight the area of emerging SCORM standards for interactive simulations. Working groups such as the IEEE SCORM and Simulation Interoperability study group are working to identify potential standards to more directly support this type of integration. This project will help further this effort.

Together, these standards, technologies and tools have the potential to make future training simulations even more accessible, adaptable, affordable, interoperable and reusable. This objective is consistent with Navy desires to make training more dynamic and engaging, more accessible, more deployable and less expensive. Although the required technologies exist and have been successfully demonstrated, the challenge is to formalize them together in an architecture that can be integrated into the ILE.

REFERENCES

BMH Incorporated, (2005) "Delta3D Game Manager Software Design Document – Version 1.0", Alion Science & Technology, BMH Operation.

Kelly, H., et al, (2002), "Training Technology against Terror: Using Advanced Technology to Prepare America's Emergency Medical Personnel and First Responders for a Weapon of Mass Destruction Attack", Federation of American Scientists.

Learning Systems Architecture Laboratory (LSAL), "The Technical Evolution of SCORM", Carnegie Mellon University.

Maskell, R., (2003), "Taking Learning to the Next Level", Military Training Technology, Volume 8, Issue 2.

Navy ILE, (2004), "Content Design, Development and Deployment for the Navy Integrated Learning Environment", Version 1.41.

Navy ILE, (2006), "Navy ILE Introduction", MPT&ECIOSWIT-ILE-INTR-1, Retrieved March 2, 2006, from Navy ILE Policy and Guidance Website, http://navyile.fedsun.navy.mil/netcile/jsp/mainframeILE.jsp

Navy ILE, (2006), "Navy ILE Technical Specifications and Guidelines", MPT&ECIOSWIT-ILE-SPEC-4A, Retrieved on March 2, 2006, from Navy ILE Policy and Guidance Website, http://navyile.fedsun.navy.mil/netcile/jsp/mainframeILE.jsp

Navy ILE, (2006), "Navy ILE PC Modeling and Simulation Guidelines, Volume 1: Overview", MPT&ECIOSWIT-ILE-GUID-4, Retrieved on February 1, 2006, from http://navyile.fedsun.navy.mil/netcile/jsp/mainframILE.jsp

Navy Personnel Development Command (NPDC), (2004) "Enabling the Navy Revolution in Training – An Overview of the Navy Integrated Learning Environment (ILE)", Version 1.1.