EXPERIENCE. RESULTS.

A WHITE PAPER FROM THE NAVY EDUCATION AND TRAINING & PROFESSIONAL DEVELOPMENT TECHNOLOGY CENTER

User Interface (UI) Interoperability for SCORM 2.0

Jason Haag
Principal Systems Engineer
Computer Sciences Corp.
Navy eLearning
Pensacola, FL
jason.l.haag@navy.mil

ABSTRACT

Although one of the primary tenets of the Sharable Content Object Reference Model (SCORM) was to promote interoperability of learning content, it still remains one of the biggest challenges today. The SCORM has been highly successfully in making the run-time communications and the learner's performance data associated with learning content, interoperable, by incorporating the IEEE 1484.11.2-2003 Standard for Learning. However, SCORM has remained silent about how a Learning Management System (LMS) can implement various technical aspects of the user interface. The SCORM has far too long dismissed several elements of the user interface as being "outside the scope of SCORM". Ignoring how the learner may interact with or access their content from an LMS (or other content delivery application) has severely and negatively impacted both the technical interoperability of SCORM content as well as the user experience of the learner. Addressing "why" the learner may interact with various elements of self-paced content is the primary responsibility of the content development team, but the SCORM should provide a specification that would present options for "how" the learner might interact with multiple user interfaces regardless of the technology medium employed to deliver the content (e.g. web browsers, mobile devices, etc) .

Some of the Department of Defense (DOD) services have had negative experiences when attempting to share SCORM content packages between their various LMS implementations primarily due to differences with both user interfaces and the Application Programming Interface (API) Implementation. The vision of plug-n-play interoperability of learning content is usually achieved only after several additional hours of modifying the content to work in a particular LMS implementation. In order to achieve adoption on a global scale, SCORM 2.0 must have a strategy to improve interoperability by standardizing the user interface controls in further support of flexibility, usability, accessibility, and durability. This paper provides a background and summary of the Navy's successes with extending the SCORM to support standardized user interface options, and further proposes creating or incorporating a new user interface interoperability specification and a recommendation for supplying a standardized API Implementation as part of the Core SCORM.

ABOUT THE AUTHOR

Jason Haag's professional interest and background is in learning technology. He is currently employed by Computer Sciences Corporation as a principal systems engineer on the U.S. Navy's eLearning program where he has been involved with implementing the SCORM for more than seven years. He is an ADL Sharable Content Object Model (SCORM) technical working group participant, represents the US Navy on the Defense Advanced Distributed Learning Action Team (DADLAT), member of the IEEE Computer Society, and IEEE Learning Technology Standards Committee (LTSC). He also operates and maintains a free SCORM resource website, CONFORM 2 SCORM, http://www.conform2scorm.com.

User Interface (UI) Interoperability for SCORM 2.0

PROBLEM DEFINITION

Arguably, the most important of the “-ilities” of the SCORM is interoperability. One of the most critical problems with previous versions of the SCORM is that it never required the implementation of a standard user interface (UI). This topic, user interface interoperability, will be the primary focus of this paper, but there is a secondary aspect of UI interoperability that will be also be addressed: the SCORM Application Programming Interface (API) Implementation. Previous versions of SCORM primarily addressed the interoperability of packaging the content and the interoperability of the Run-Time Environment (RTE) CMI data model. Unfortunately, a standardized approach for supplying both the 1) UI for launching the content and the 2)API Implementation were not addressed in SCORM, and consequently were left to be supported in proprietary methods by competing LMS vendors in the eLearning marketplace. As a result, interoperability has not been fully realized by consumers of SCORM. In addition, previous versions of SCORM did not account for inevitable changes to the primary delivery medium: the web browser.

Inconsistent User Interfaces

The US Navy has collaborated with the US Marine Corps (USMC), the US Coast Guard (USCG), and various other international military services and organizations in contact with the US Navy for the purpose of sharing their existing content libraries. Within the DoD community there are several web-based mandatory courses that often reflect the same subject matter so making content sharable was the one of the most obvious advantages that SCORM promised to deliver. While the US Navy has experienced some degree of success in sharing content with other DoD services it did not initially happen without a great deal of additional modification and manual labor required to first make the content interoperable. The concept of making content “plug-n-play” was a primary goal of SCORM, but is has not been easy for content developers to achieve primarily because the UI for launching the content was not implemented in a standard manner by LMS vendors. The aforementioned technical modifications made to the content are usually attributed to making changes such that content will behave in a consistent manner by uniquely different LMS implementations. These launch behaviors and other UI aspects are inherent properties of the web browser such as the width and height of the browser, the browser toolbar, resizing, and the area where the content is first accessed within the browser window hierarchy (e.g. frameset, new window, etc.). These aspects of UI severely impact the interoperability and must be addressed in SCORM 2.0.

Differing API Implementations

In addition to addressing the UI challenges associated with the web browser, SCORM 2.0 must supply a standardized methodology for supplying the API Implementation. While the SCORM actually provided example approaches to supplying the API Implementation, LMS vendors did not often use the same programmatic strategies or technologies and as a result, interoperability and sharing of the content was not easily achieved without making major programmatic changes to the content itself. While SCORM eventually improved run-time interoperability by removing all of the SCORM 1.2 optional aspects of implementing the CMI data model, the various technologies employed by LMS vendors remained non-standardized. The introduction of Sequencing and Navigation further complicated things and often made each LMS vendor’s API Implementation even more unique and proprietary. Many LMS vendors often used Java and/or other technologies to develop their own API Implementation, and provided the API in inconsistent locations within the browser window hierarchy. The resulting approach to accommodate this inconsistency was to usually make changes to the content as well as the sample JavaScript-based API wrapper provided by the Advanced Distributed Learning (ADL) initiative. Java remains a technological challenge with the US Navy and USMC because of the tightly secured workstations that a majority of the learners use for accessing content. While Java is often touted as being platform-independent there still remain some accessibility and configuration challenges to its usage within the Navy. The prospect of not supplying a standardized API implementation will continue to provide further challenges to the interoperability of content if not addressed in SCORM 2.0. The code base and technology employed for the API implementation in SCORM 2.0 must be both ubiquitous and controlled in order to withstand inevitable changes to web browsers, operating systems, and other technological variables that are paramount to the delivery, portability, and interoperability of the content. With the recent advances in mobile standardization and best practices provided by the World Wide Consortium (W3C) Mobile Web Initiative, device independence should be another key consideration. A standardized API Implementation should utilize technology that would be supported by any delivery platform and accessible on any device. Finally, the code base for a standardized API Implementation could be centrally controlled by a LETSI working group or IEEE working group so that both new technological innovations as well as changes to existing delivery platforms and devices could be uniformly supported.

USE CASE

The US Navy deployed their first instance of a SCORM-conformant LMS during the era of SCORM 1.2 (circa 2001). Many challenges to interoperability were first recognized during the testing of SCORM-conformant content that had only been previously tested in the ADL Conformance Test Suite (CTS) and Sample Run Time Environment (RTE). The Sample RTE provided content developers with an interface that differed significantly from the actual interface provided by the Navy's LMS. Aside from subtle differences in the API Implementation, the UI for the Sample RTE launched the content in a frameset. In comparison, the Navy's LMS provided an interface using several opener windows prior to providing the frameset where the API instance could be located. In addition, there were several obvious inconsistencies between the Sample RTE and the Navy's LMS in regards to the width, height, and other browser window attributes. The same challenges to interoperability became more evident when the USMC attempted to share a course titled, "Driving for Life" with the Navy. During the era of SCORM 1.2 there was no support or direction for navigational elements associated with the UI. As a result, some LMS vendors implemented their own proprietary navigation elements for their UI. Alternatively, some LMS vendors provided no navigation support at all. This inconsistency left many content developers with the responsibility to programmatically control the internal navigation functions associated with "exiting" the course and closing the browser window. This finding was the first of many other interoperability challenges for SCORM 1.2 (e.g. rollup, scoring, etc.). With the release of SCORM 2004, some of the interoperability issues associated with UI navigation such as "exiting" were addressed, but several other UI challenges still remained constant. As the Navy's learner population and content library continued to grow, so did the requirement to support a wider content distribution strategy. The requirement to provide equal access to learning opportunities throughout the Fleet forced the Navy to plan for content to be deployed to multiple learning management systems and their respective environments such as classrooms, classified/SIPR, ships and submarines, and disconnected/offline applications. Interoperability soon became the US Navy's primary goal when deploying SCORM content. As a result of these UI interoperability challenges, new deployment requirements, and inflexible support of SCORM 2004 from the LMS, the US Navy sought out a solution from Rustici Software, LLC. The Navy first learned of the Rustici SCORM Engine (RSE) product after several discussions and collaboration with the USMC. The RSE was being used by the USMC's LMS and had been offered to the US Navy to fill the UI interoperability void that was inherently, and unfortunately part of the SCORM. The RSE product provided an alternative to organizations such as the US Navy that were left with LMS vendors that provided proprietary and inflexible implementations of the SCORM. The RSE product acts as a third-party solution that handles all aspects of the SCORM RTE and provides the LMS with a robust integration layer allowing the LMS to focus on LMS functionality while letting the RSE take care of the complexities and interoperability challenges of SCORM. The RSE provides an enhancement to the SCORM by exposing UI properties that can be set for each course to control precisely how it is delivered to the user. These UI properties can be specified by the content developer in the content package via an extension to SCORM metadata. An example of highlighting some of the UI properties provided as a RSE extension to the IEEE LOM are provided in Figure 1 using a sample SCORM metadata file. This sample is not intended to represent all of the metadata elements or the complete RSE XML binding, but is provided as an example to show those elements that provide support for UI interoperability via the RSE. The sections highlighted in yellow in Figure 1 indicate where the SCORM has been extended in the IEEE LOM metadata file.

```
<?xml version="1.0"?>
<lom xmlns="http://ltsc.ieee.org/xsd/LOM" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://ltsc.ieee.org/xsd/LOM lomcustomco.xsd" http://www.scorm.com/HSTMConfiguration.xsd
HSTMConfiguration.xsd">
        <general>
                <identifier>
                  <catalog>NEL</catalog>
                  <entry>CNL-DFL-1.0</entry>
                </identifier>
                <title>
                        <string language="en">Driving for Life</string>
                </title>
                <language>en</language>
                <description>
                        <string language="en">The purpose of this course is to facilitate the understanding of knowledge necessary to
ensure safe driving practices for Marines and Sailors. Specifically, this course will focus on the driving scenarios that cause a high number of
deaths in the target population and how to react to potentially dangerous driving situations.</string>
                </description>
                <keyword>
                        <string language="en">Driving</string>
                </keyword>
                <keyword>
                        <string language="en">Safe Driver</string>
                </keyword>
                <keyword>
                        <string language="en">Driver Safety</string>
                </keyword>
        <technical>
        <HSTMConfiguration xmlns="http://www.scorm.com/HSTMConfiguration.xsd">
        <controls>
                <showFinishButton>yes</showFinishButton>
                <showHelp>yes</showHelp>
                <showProgressBar>yes</showProgressBar>
                <showCourseStructure>yes</showCourseStructure>
                <courseStructureStartsOpen>yes</courseStructureStartsOpen>
                <showNavBar>yes</showNavBar>
                <showTitleBar>yes</showTitleBar>
                <statusDisplay>combined</statusDisplay>
        </controls>
        <appearence>
                <courseStructureWidth>300</courseStructureWidth>
                <displayStage>
                        <width>1024</width>
                        <height>768</height>
                        <fullscreen>no</fullscreen>
                </displayStage>
        </appearence>
        <behavior>
                <disableRightClick>no</disableRightClick>
                <preventWindowResize>no</preventWindowResize>
                <launch>
                 <sco>frameset</sco>
                 <player>new window</player>
                 <wrapScoWindowWithApi>yes</wrapScoWindowWithApi>
                </launch>
        </behavior>
        </HSTMConfiguration>
        </technical>
</lom>
```

Figure 1: Sample Metadata File with User Interface (UI) Interoperability Controls Supported by the RSE

The RSE extension elements are placed within the technical element of SCORM metadata in a container element named “HSTMConfiguration”. This element is also the name of the RSE XML Schema (xsd) describing the XML binding of these parameters (which are used to validate the format of the extensions). A namespace declaration is made at the top of the metadata file as well as adding the “hstm” prefix on the HSTMConfiguration element. By providing content developers with complete control over how the content should be launched from the LMS, the challenges associated with UI interoperability become non-existent. Navy courses that were originally designed to only launch each Sharable Content Object (SCO) in a frameset window could now work with any LMS integrated with the RSE such as the USMC's LMS. As a result of implementing the RSE and extending SCORM metadata, the US Navy is now able to provide a consistent UI experience for learners of the US Navy's many LMS environments in addition to sharing content with the USMC.

STAKEHOLDERS

The stakeholders that would benefit most from implementing the proposed solution would be all consumers and supporters of the SCORM including, but not limited to the following: learners, instructors, LMS vendors, content authoring application developers, content developers, content providers, graphic designers, instructional designers, technical implementers, and LMS administrators.

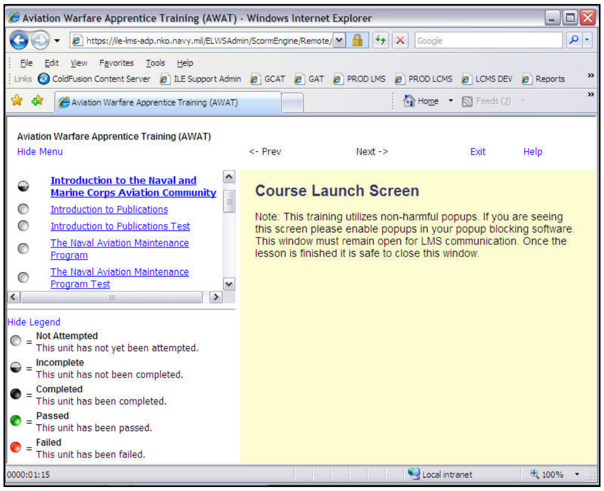
PROPOSED SOLUTION

A combination of solutions would be required to solve the two UI interoperability problems defined in this whitepaper. The first solution is to supply a standardized API Implementation as part of the Core SCORM. Addressing the UI aspects concerned with only the web browser is not enough. As long as LMS vendors and other content delivery-based applications are afforded the option to develop their own proprietary API Implementation, the interoperability problem in SCORM will continue to exist. SCORM was successful in standardizing the content so why not also the API Implementation? Supplying the SCORM community with a standard API Implementation should be addressed before entertaining any other ideas or proposals of replacing or enhancing the content aggregation model, sequencing and navigation, metadata, and/or the CMI data model.

SCORM 2.0 must have a strategy to substantially improve interoperability through standardized UI controls that provide optimal flexibility. The second part of the combined solution to the UI interoperability problem is to create a UI interoperability specification as part of the Core SCORM. Support for this "flexibility" aspect of the UI could actually be part of the SCORM 2004 specification today. In fact, the current ADL content packaging schema could even be extended to provide a basic level of flexibility by adding some fundamental support parameters of the web browser such as the default window size and whether the content should be delivered in a new window or a frameset. However, since it should be expected that SCORM will possibly be supported by more than just PC web-browsers in the future a more comprehensive specification and XML binding to support additional aspects of UI interoperability would have to wait until SCORM 2.0. If one thinks about interoperability on a larger scale than just web browser flexibility, it's actually impacted by and many times directly related to several additional concerns involving usability, accessibility, and durability. Addressing all of these concerns through a UI interoperability specification will require a collaborative and dedicated working group effort led by LETSI.

UI Interoperability Based On Usability

Usability is referred to in this whitepaper as applying best practices for the information architecture, functionality, internationalization, and designing visual elements of a specific UI. Usability should be considered as part of the proposed UI interoperability specification for several of the obvious reasons, but one particular aspect that comes to mind is the method for providing visual indicators of the learner's progress. This concern has been discussed in the past among the ADL SCORM Technical Working Group (TWG) on their forums and a proposal for an Activity Tree Rendering (ATR) was previously submitted (Ostyn, 2007). The Navy addressed this ATR concern in the RSE through the element named "<statusDisplay>" in Figure 1. See Figure 2 below for how the Navy has implemented the graphics (visual indicators) associated with the <statusDisplay> element as part of the ATR displaying the learner's progress.


Aviation Warfare Apprentice Training (AWAT) - Windows Internet Explorer
https://ile-lms-adp.nko.navy.mil/ELWSAdmin/ScormEngine/Remote/
Google
File Edit View Favorites Tools Help
Links ColdFusion Content Server ILE Support Admin GCAT GAT PROD LMS PROD LCMS LCMS DEV Reports
Aviation Warfare Apprentice Training (AWAT)
Home
Feeds (J)
Hide Menu
<- Prev
Next ->
Exit
Help
Hide Legend
0000:01:15
Local intranet
100%

Aviation Warfare Apprentice Training (AWAT)

- ◒ Introduction to the Naval and Marine Corps Aviation Community
- ○ Introduction to Publications
- ○ Introduction to Publications Test
- ○ The Naval Aviation Maintenance Program
- ○ The Naval Aviation Maintenance Program Test

Course Launch Screen

Note: This training utilizes non-harmful popups. If you are seeing this screen please enable popups in your popup blocking software. This window must remain open for LMS communication. Once the lesson is finished it is safe to close this window.

○ = Not Attempted
This unit has not yet been attempted.

◒ = Incomplete
This unit has not been completed.

● = Completed
This unit has been completed.

🟢 = Passed
This unit has been passed.

🔴 = Failed
This unit has been failed.

Figure 2: Screen Capture of User Interface (UI) Provided for the Launched Content. The Legend of Visual Indicators is Located in the Bottom Left.

UI Interoperability Based On Accessibility

While the Navy's LMS and UI for the ATR provides good usability practices and visual indicators to display the user's progress it does not provide internationalization features, and more importantly does not fully cover web accessibility. The World Wide Web (W3C) Consortium states that web accessibility encompasses all disabilities that affect access to the Web, including visual, auditory, physical, speech, cognitive, and neurological disabilities. Of equal importance is that web accessibility also benefits people without disabilities. The Navy's UI for the ATR provides basic support for web accessibility such as ALT text and keyboard accessible functions, but the visual indicators are based on colors and could potentially be confused by a color-blind user. Support of web accessibility through a UI specification would lay the foundation optimal adoption of SCORM 2.0. In addition, if web accessibility were fully supported by a UI interoperability specification in SCORM 2.0 then it would inherently benefit from other W3C recommendations associated with web accessibility such as the W3C Mobile Web Initiative's (MWI) Mobile Web Best Practices. Figure 1 represents elements that allow the content developer to control the width and height for their content, but these are static parameters that currently only apply to a PC-based web browser. Most of the Navy's SCORM content as well as the Navy's LMS are only functional from a web browser on a PC. One might assume that this is also likely the case for most other consumers of SCORM. If such W3C recommendations were supported in a UI specification in SCORM 2.0 then content developers could ultimately provide the learners with advanced options for alternate versions of their content for display on multiple devices and browsers.

UI Interoperability Based On Durability

Durability in SCORM today implies that the learning systems of the future will be compatible with SCORM content of today. For the most part, SCORM has been very successful in obtaining this goal today, but the pillar of durability will need to be expanded to accommodate for more than just learning systems that utilize PC-based web browsers. Content developers should not have to modify learning content when web browsers change, but that has not always been the case. Proactive changes to the security models of web browsers as well as the addition of new features (e.g. pop-up blockers, tabular browsing, etc.) have inadvertently broken some SCORM content. While the key technology behind SCORM (ECMAScript, XHTML, and XML) remains durable, the user interface designed for the many SCORM courses are often not. In fact, most SCORM 2004 content today still contains embedded navigation that hinders interoperability because standardizing the UI has not been addressed. The proposed UI Interoperability specification could be periodically updated as needed to accommodate for custom UI experiences as well as resolve unforeseen technological changes to the various delivery platforms and devices (e.g. web browsers, mobile browsers, etc.). This proposal does not imply changing the technology behind SCORM, but instead adding support for a UI interoperability specification to support potential changes to the technology changes made to the software applications responsible for rending the content. By ensuring a clean separation of the UI from the LMS, the content will less likely become a victim of technological change and will degrade more gracefully.

INTEGRATION & OTHER ISSUES

The integration of a UI Interoperability specification could be achieved by incorporating support of current W3C recommendations as well as building upon the lessons learned from the use case provided in this whitepaper. Given the multitude of mobile and web browsers that can now deliver web content, the effort involved with testing the proposed solution could be quite intensive. Former versions of the SCORM were supported by a large support community and received resource funding from the DoD. While funding and resources are not a technical issue per se, they will most likely become one of the biggest hurdles to realizing many of the ideas and technical proposals for SCORM 2.0. It should be recommended that the proposed solution be governed by a LETSI working group and that all development work be made available as an open-source project. The development of a standardized API Implementation would be a more challenging proposition, but one that could potentially provide unlimited interoperability benefits to the SCORM. Conceptually, these two proposed efforts should be planned and integrated into the SCORM in conjunction with one another since they both address issues related to interoperability.

SUMMARY

In order to achieve adoption on a global scale, SCORM 2.0 must have a strategy to significantly improve content interoperability by defining standardized user interface controls and options that will allow support for flexibility, usability, accessibility, and durability. In addition, SCORM 2.0 should explore an open-source approach led by the LETSI SCORM community in the development of a standardized API Implementation. These two critical aspects of interoperability should be addressed before entertaining any other ideas or proposals of replacing or enhancing the content aggregation model, sequencing and navigation, metadata, and/or the CMI data model. Current versions of the SCORM have been extended by the Navy to resolve some interoperability challenges to SCORM. However, the interoperability challenges still exist for many organizations and consumers of SCORM today. LMS vendors do not often use the same programmatic strategies or technologies for supplying the UI and API Implementation and as a result, interoperability and sharing of the content is not easily achieved without making major programmatic changes to the content itself. By adopting the proposed UI Interoperability specification and a standardized API Implementation, SCORM content of the future could be made more interoperable and shared by more than just the LMS applications of today.

ACKNOWLEDGEMENTS

The author would like to acknowledge the support of the sponsoring commands and organizations: Navy Education Training and Professional Development Technology Center (NETPDTC), the Department of Navy (DoN) Program Executive Office for Enterprise Information Systems (PEO (EIS)), Computer Sciences Corporation (CSC), Rustici Software LLC, and the federation for Learning Education and Training Systems Interoperability (LETSI).

REFERENCES

Advanced Distributed Learning (ADL) Initiative, SCORM Documentation (n.d.). Retrieved August, 2008 from: http://www.adlnet.gov/scorm/

Mobile Web Best Practices 1.0, W3C Recommendation, Jo Rabin, Charles McCathie Nevile eds., 29 July 2009. Available at: http://www.w3.org/TR/2008/REC-mobile-bp-20080729

Ostyn, C., Proposal for Activity Tree Rendering . Retrieved July 2008 from: the ADL Technical Working Group: https://community.adlnet.org/sites/SCORMTechWG/

Rustici Software's SCORM Engine (RSE) Metadata Extension, Rustici Software Sofware, LLC., 3326 Aspen Grove Drive Suite 304, Franklin, TN 37067: http://www.scorm.com

Web Accessibility Initiative (WAI)’s Web Content Accessibility Guidelines (WCAG) 2.0, W3C Proposed Recommendation, Available at: http://www.w3.org/TR/2008/REC-mobile-bp-20080729