4600 South Ulster St, Suite 450
Denver, CO 80237
(303) 483-2800
www.medsimulation.com

# Simplifying SCORM While Keeping it Feature Rich

*By Yuanming Chen, Shared Technology Manager*
*Mary Goble Thomas, e-Learning Developer*
*Wes McNeil, e-Learning Developer*
*Vadieh Rahmani, Multimedia Specialist*

**August 12, 2008**

# Contents

## 1 Abstract

During implementation of several of our courses to meet the SCORM 2004 specifications, we noted several things that we believe would enhance the SCORM, lead to broader acceptance of it, and promote greater interoperability between vendors. This paper describes several of our proposals, including dedicated peer-reviewed application programming interfaces (APIs), simplifying the CMI data model, and maintaining backward compatibility for legacy content. We acknowledge the importance of the SCORM and offer the following recommendations for consideration.

## 2 Problem Definition

E-learning is changing and encompassing many new domains. For example, medical simulation is a training method in which medical professionals practice procedures in lifelike circumstances. Among features leading to effective learning, “providing feedback to the learner” was the most frequently cited benefit of the simulation exercise.[1] In endovascular medical simulation, key behaviors such as various stages of translation and rotation in catheter manipulation, balloon inflation and stent deployment should be tracked in real time. Other key findings to record would be amount of time needed to treat adverse events.

Unlike most e-learning courses, medical simulation applications are typically not deployed on the web due to constraints of performance and complexity, but they should be considered e-learning products nonetheless. But the current SCORM is web-centric, making it difficult for medical simulation companies and game companies to take full advantage of an industry standard that is meant to “provide access to the highest quality learning and performance aiding that can be tailored to individual needs, delivered cost-effectively anytime and anywhere.”[2]

[1] Issenberg SB, Features and uses of high-fidelity medical simulations that lead to effective Learning: a BEME systematic review, Jan 2005.

[2] SCORM 2004 3rd Edition, Sharable Content Object Reference Model, Overview, November 16, 2006

## 3 Use Cases

In a medical simulation education environment there are a number of use cases to consider, for both consumers and suppliers of e-learning:

| Use Case | Actors | Systems | Goals |
|---|---|---|---|
| Simulation Learning | Physicians, nurses, technicians, medical students | Medical simulation systems | Learn new procedures/devices and improve existing skills with risk-free practice |
| Web Certification | Physicians, nurses, technicians | Training website | Obtain or renew a certificate |
| Didactic Learning | Physicians, nurses, technicians, medical students | Training website, medical simulation systems, CD, external LMS | Learn new areas, improve existing areas of knowledge |
| Administrator Data Review | Hospitals, professional medical societies, academic entities, regulatory and credentialing organizations | Reports created by training website, external LMS, and medical simulation systems | Evaluating effectiveness of learning, evaluating performance, assessing readiness of staff |
| Supplier Data Review | Content developers, instructional designers, e-learning developers, software developers | Reports created by training website, external LMS, and medical simulation systems | Evaluating effectiveness of e-learning courses, improvement of courses |
| Student Data Review | Physicians, nurses, technicians, medical students | On-screen reports created by training website, external LMS, and medical simulation systems, during training session | Learning from mistakes |
| Demonstration | Simulation company, clients, physicians, nurses, technicians | Medical simulation systems | Demonstrate a device and/or procedure to others |
| Course Development | Instructional designers, subject matter experts, Flash developers, simulation developers, HTML developers | Content authoring software/languages, language-specific APIs, SCORM APIs, middleware, distribution mechanisms | Create meaningful learning experiences on time and within budget |

# 4 Stakeholders

Our proposed solution would expand SCORM's reach into new domains such as medical simulation. The following is a list of stakeholders who would be interacting with each other in the medical simulation domain.

- Suppliers
  - E-learning developers
  - Content developers
  - Simulation software developers
  - Instructional designers
- Distributors
  - Internet service providers
  - Learning management system providers
  - Software retailers
- Consumers
  - Physicians (Expert clinicians, fellows and residents)
  - Nurses
  - Medical technicians
  - Medical students
  - Other health professionals
  - Medical device manufacturers
  - Health care systems and hospitals
  - Professional medical societies
  - Academic entities
- Regulatory and credentialing organizations
- Patients

# 5 Proposed Solution

## *5.1 Requirement Analysis*

One of the main goals of SCORM 2.0 is to "be an interoperability model that, like its predecessor, can be used strategically across market sectors and geographical regions".[3] The starting point of our technical solution is to identify key success factors for SCORM 2.0 that will address the needs of different stakeholders in those market sectors. A guiding philosophy is to "make simple things simple."

### 5.1.1 Lower barrier of entry

An overly complex e-learning model is often daunting to content developers, who might not have the time or willingness to learn every nuance (e.g., CMI data model, imsmanifest XML and navigation systems) required to achieve SCORM conformance.

[3] http://www.letsi.org/letsi/download/attachments/4751660/LETSI+White+Paper+Solicitation+on+SCORM+31May08+FINAL.pdf

### 5.1.2 Empower developers with languages of their choice

The language of choice in the current SCORM is JavaScript.[4] Under certain circumstances, JavaScript might not be an ideal language. For example, Flash developers prefer to use ActionScript, while simulation and game developers tend to choose Python, Lua or Ruby. Although various third-party API wrappers exist to simplify SCORM communication,[5,6] they often create new problems (e.g. non-standardized syntax).[7]

It should be noted that JavaScript was undoubtedly a good cho ice when SCORM was first introduced, because SCORM's original goal was to "enable interoperability, accessibility and reusability of web-based learning content,"[8] and JavaScript happened to be the de facto language for all web browsers. But as SCORM 2.0 is poised to expand into areas like games and simulation, JavaScript, an interpreted language, will struggle to meet the high-performance requirements for SCORM communication in those applications.

### 5.1.3 Help developers to write reliable code

Writing reliable SCORM code is not easy with the current SCORM specifications. Although SCORM's API functions are very simple, the underlying CMI data model is intimidating for even expert developers (e.g. cmi.interactions). The loss of static type safety (e.g. parameters for SetValue() function can only be of string type) means many errors can only be detected at runtime.

From an architectural perspective, software designers need to carefully evaluate the pros and cons of hiding all complexities of SCORM under the umbrella of a do-it-all interface. Some negative consequences of this approach are intellectual overhead and inefficiency.[9]

### 5.1.4 Develop flexibility and extensibility

Not all stakeholders have the same needs. For example, some people are looking for tighter security features in SCORM while others might not care. With the passage of time, even the same party could have different expectations for SCORM. A more challenging problem is how to keep SCORM relevant in an ever-changing world.

In the new SCORM design, while great effort should be dedicated to make SCORM simpler for entry-level users, we should not ignore the need to build flexibility and extensibility into the SCORM model in order to reap the benefits of the latest technologies. It could be argued that bleeding-edge features have been the driving forces behind SCORM's evolvement over the years.

Using the words of Albert Einstein, we could rephrase our guiding philosophy for SCORM 2.0 as the following: "**Keep it as simple as possible, but not simpler**."

[4] SCORM 2004 3rd Edition Sharable Content Object Reference Model, Run-Time Environment, 3.1.1
[5] http://pipwerks.com/lab/downloads.php
[6] http://www.academiccolab.org/libscorm
[7] http://pipwerks.com/journal/2008/05/22/extending-the-scorm-wrapper-and-actionscript-classes/
[8] http://www.adlnet.gov/scorm/
[9] Modern C++ Design, Andrei Alexandrescu, February 1, 2001

#### 5.1.5 Support non-web deployment

If the new SCORM supports non-web deployment, the beneficiaries will not only be simulation and game companies, but also web content providers, since web content will be deployable in non-web environments (e.g., DVDs, desktop computers, and proprietary LMSs that do not support SCORM).

By actively expanding the reach of SCORM's targeted audiences, LETSI has the potential to radically change the landscape of the e-learning industry. For key stakeholders who aim to shape the direction of SCORM specifications, the importance of incorporating medical simulation components into SCORM should not be underestimated, since improved training experiences would have far-reaching impacts on patient safety.

### *5.2 Architectural Analysis*

With a better understanding of the requirements, we will go on to evaluate two design choices that have the potential to address those requirements:

#### Approach 1: Evolve SCORM data models, but keep SCORM API intact

This was the traditional approach taken by ADL to evolve SCORM in the past. Over the years, ADL added many new features to SCORM; for example, content packaging profiles were added in SCORM 1.2, Sequencing in SCORM 1.3 and Navigation in SCORM 2004. Compared to the expansion effort of CMI data models, the SCORM API underwent far fewer changes. Here is a comparison of APIs between SCORM 1.0 and SCORM 2004:

**Figure 1: Historic changes of SCORM API**

| **SCORM 1.0** | **SCORM 2004** |
|---|---|
| return_value = LMSInitialize(parameter) | return_value = Initialize(parameter) |
| LMSFinish () | return_value = Terminate(parameter) |
| return_value = LMSGetValue (parameter) | return_value = GetValue(parameter) |
| LMSSetValue (parameter_1, parameter_2) | return_value = SetValue(parameter_1, parameter_2) |
| LMSCommit (parameter) | return_value = Commit(parameter) |
| return_value = LMSGetLastError() | return_value = GetLastError() |
| return_value = LMSGetErrorString(parameter) | return_value = GetErrorString(parameter) |
| return_value = LMSGetDiagnostic(parameter) | return_value = GetDiagnostic (parameter) |

Why is this seemingly innocuous approach of keeping the API intact while changing the CMI data model unsuitable for SCORM 2.0? We will point out two flaws inherent in this approach:

1. Incompatible data models between versions can cause major headaches for SCORM developers. Since impacts on data model changes cannot be evaluated during compile time, developers have to do exhaustive tests to verify that their old code still works

under the new SCORM. Philip Hutchison showed two such examples in his article "Extending the SCORM wrapper and ActionScript classes":[10]

```
//SCORM 2004
scorm.set("cmi.lesson_location", "page1");
scorm.set("cmi.completion_status", "incomplete");

//SCORM 1.2
scorm.set("cmi.core.lesson_location", "page1");
scorm.set("cmi.core.lesson_status", "incomplete");
```

2. When a new SCORM version comes out, content developers have to choose either to learn nuances of the new standard or not to migrate to the new specifications. Both choices could be bad choices.

### Approach 2: Expand SCORM API, and simplify the CMI data model

This approach is inspired by Boost C++ Libraries,[11] a collection of over 60 free peer-reviewed portable C++ source libraries. Using the concept of Boost, we will design SCORM 2.0 with the following design choices:

1. Instead of expanding the CMI data model to meet the needs of new market sectors (like medical simulations and games), we will design dedicated APIs for those domains. In the foreseeable future, the SCORM would contain many specialized APIs in different functional areas.

2. The CMI data model will be simplified. If necessary, equivalent functionalities will be replaced by specialized APIs. In Philip Hutchison's article "What do you want *your* SCORM to do?",[12] he proposed to add four APIs:

    - A mandatory API for simple course-to-LMS communication
    - An optional reusability/shareable content mechanism
    - An optional navigation API
    - An optional quiz/exam API

3. APIs are peer-reviewed and will be finalized with the approval of a reviewing board.

4. APIs must conform to ECMA standard. This will allow content developers to use their preferred scripting languages in the software development.

5. A review board will ensure that APIs are always backward compatible.

This software design will bring many benefits to all SCORM stakeholders:

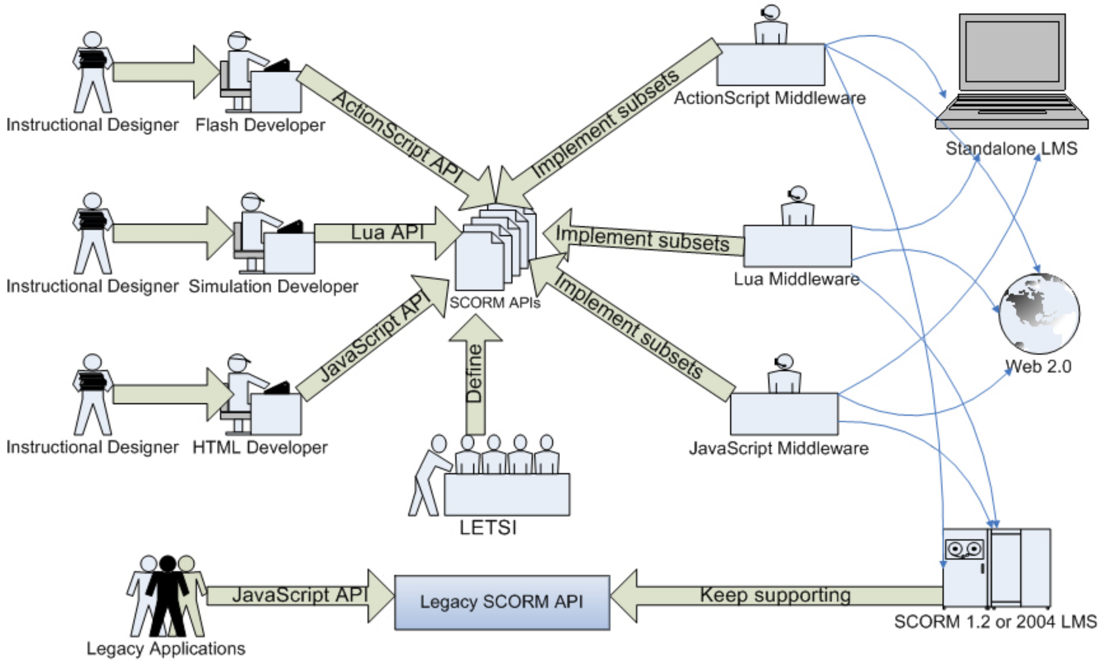
[10] http://pipwerks.com/journal/2008/05/22/extending-the-scorm-wrapper-and-actionscript-classes/
[11] http://www.boost.org/
[12] http://pipwerks.com/journal/2008/06/22/what-do-you-want-your-scorm-to-do/

1. Developers will have a better chance to write simpler and more reliable code. Compilers can catch more errors during compile time.

2. Clean and simple interfaces would hide the implementation details and save SCO developers from having to know about those implementations (e.g. CMI data model). It makes it possible to offer different implementations for the same interface. APIs serve as an abstraction layer and make it possible to deploy the same content into different hosting environments, including non-web environments.

3. Since new APIs can be added to the current collections without impacting existing APIs, this proposal would achieve the design goal of adding flexibility and extensibility to the SCORM.

## 6 Integration and Technical Issues



The diagram shows how different stakeholders interact with each other in the proposed SCORM 2.0 framework. A noticeable difference from previous frameworks is the addition of SCORM middleware providers, who offer implementations of APIs defined by LETSI. It is worth pointing out that a middleware provider doesn't need to implement all existing APIs. For example, medical simulation middleware providers might only offer medical simulation-related API implementations, and their offerings will be more geared toward meeting the real-time requirements for high-performance simulations. Other middleware providers could be specialized in offering middleware implementations with enhanced security features.

For developers who still prefer to use the legacy SCORM API and CMI data model, they can continue to use their traditional methods of communicating with a SCORM-compliant LMS. They could also consider a migration plan to take advantage of the features in our proposal, including:

- Type safety
- Write-once-deploy-everywhere
- Performance gain
- Code simplicity

Since SCORM middleware plays an important role in the proposed SCORM infrastructure, LETSI should encourage all middleware providers to offer SCORM implementations as free downloads. LETSI should also design a website that will make it easy for content developers to search the desired middleware. In addition, the website should offer thorough and clear API documentation and relevant code examples.

We also suggest that middleware providers should consider offering value-added services like basic data reporting and automated testing to help content developers write better software.

## 7 Existing Implementations

Medical Simulation Corporation successfully implemented the software infrastructure mentioned above. By developing an API and the corresponding middleware component, we are able to produce one version of software that can be deployed on multiple platforms in order to meet different training requirements. Since the complexities of SCORM are hidden, developers are writing code that is more expressive, more maintainable and more portable.

## 8 Summary

As mentioned by LETSI, SCORM 2.0 is intended to be an interoperability model that can be used strategically across market sectors and geographical regions. In order to achieve such a model, we propose that discrete, peer-reviewed APIs be designed for different market segments. These specialized APIs would need to be backward compatible. We would also like the CMI data model to be simplified, and developers to be given the option to use their language of choice for SCORM communication. We believe the addition of specialized APIs will better empower SCO developers to write portable, reusable, and reliable software. The proposed use of middleware will allow for a feature-rich environment, enhancing the already well-established and effective SCORM.