SYBERWORKS
Learn Anytime, Any Place

411 Waverley Oaks Road
Building 3, Suite 319
Waltham, MA 02452
www.syberworks.com
1-888-642-7078

SCORM and the Learning Management System (LMS)

By Stuart Campbell, Director of Software Development for SyberWorks, Inc.

What actually is *SCORM*? SCORM, **S**hareable **C**ontent **O**bject **R**eference **M**odel, is a standard for web-based e-learning that has been developed to define communication between client-side content and a runtime environment. In the context of this article, the client-side content would be the course launched by a student and the runtime environment would be a Learning Management System. The SCORM standard has undergone several version releases from *SCORM v1.0* in 1999 to the *SCORM 2004 3rd Edition* in 2006. The purpose of this article is to look in a little more detail at the communication flow between the LMS and a course that is **SCORM v2.1 compliant**. The intention is not to review every possible communication but just the main calls. We will look at SCORM in the context of a SCORM course, i.e. a course that has been developed and built to communicate with a Learning Management System using the SCORM standard.

First of all, let me define a few terms that I will use and that you may hear in reference to SCORM courses:

- ***API*** (**A**pplication **P**rogramming **I**nterface)
  The SCORM standard has a defined *API*. The term *API* is not unique to SCORM but is a well known term in application development. The SCORM API is a documented set of messages that are sent between the course and the LMS. The message may require a response so that the sender of the message is guaranteed that the message has been received and processed appropriately.
- ***Manifest***
  The manifest is the master file for the SCORM course that contains references to all other files that make up the complete content of the course.
- ***Single SCO***
  A single SCO is the most basic and most common form of a SCORM course. All files that make up the course, regardless of the number of lessons in the course, are packaged as one SCO unit. When the course is launched within the Learning Management System, it communicates with the SCO.
- ***Multiple SCO***
  A multiple SCO is where a group of files within the course can be treated as an individual SCO but packaged together with other SCOs to comprise one course. For example, the files that comprise each lesson of a SCORM course can be grouped together so that each lesson is a SCO in its own right and all the SCOs will comprise the one course.

Course Installation

For the course to be available in the Learning Management System, the SCORM package needs to be installed on the system. The method of installation will vary from LMS to LMS but will typically go through the following steps:

411 Waverley Oaks Road
Building 3, Suite 319
Waltham, MA 02452
www.syberworks.com
1-888-642-7078

1. The manifest file will be located and interpreted.
2. The course identified in the manifest will be created in the Learning Management System and the course properties will be populated with the values defined in the manifest. These properties may include course name, creator, etc.
3. The lessons of the course will be created in the LMS and the lesson properties will be populated with the values defined in the manifest. Lesson properties would include information such as the path to the lesson file, passing grade, etc.

The Learning Management System should provide a facility to overwrite an already existing SCORM course so that if updates are made to the existing course, an installation will deploy those changes.

Course Communication

Initialization

When the course is launched within its run time environment, which for our purposes is the LMS, it will detect the existence of the API and attempt to make initial communication with the Learning Management System. This is a one-time handshake to ensure that communication has been established. The SCORM course is responsible for locating the API in the child browser session that it has launched in from the parent. The API is typically deployed using JavaScript. The API call that the course uses to do this is the LMSInitialize call. When the Learning Management System receives this API message call from the course, the LMS will initialize and respond to inform the course that communication has been established. An *object model* has been created in memory and the properties have been initialized to default values.

Intermediary Course Communication

Once the course has been launched and initialized with the LMS, the interaction between the two is determined by the course. It will use LMSGetValue and LMSSetValue API message calls to retrieve or populate data in the object model. Bear in mind that the object model exists in memory and has not been sent to the Learning Management System.

When the course needs to write the data so it exists in the database, it will send the API message LMSCommit. The LMSCommit will be received by the Learning Management System and the data stored to the LMS. The course developer will determine when an LMSCommit is required and it will be sent to the LMS when the event is activated. This could be at the end of each lesson or it could be at the end of the course.

The LMSCommit is an optional message and does not require a response from the Learning Management System.

Finalization

411 Waverley Oaks Road
Building 3, Suite 319
Waltham, MA 02452
www.syberworks.com
1-888-642-7078

When the course is completed, it will send an LMSFinish API message to the LMS. This may be triggered by an event in the course that activates the LMSFinish or it may be on the Unload event of the HTML page when the user closes the browser. The LMSFinish indicates to the LMS that the user has ended the course and any further communication between the two will be rejected. The LMSFinish also initiates the Learning Management System to store any data that has been populated in the object model and complete the unloading of the course.

The key difference between the LMSCommit and LMSFinish API message calls is that the latter will prevent further communication with the course.

The LMSFinish is a mandatory message and does not require a response from the LMS.

Learning Management System Considerations

When the LMSFinish is received by the LMS, there are several scenarios that may need to be taken into account. Some of these may be the product of errors in course development but are scenarios that have been experienced in interfacing with a SCORM compliant course:

1. The SCORM course may indicate the student has failed a lesson but the score passed from the SCO may be greater than the lesson passing grade that was installed from the manifest.
2. The SCORM course may indicate a lesson status for the user and pass the score of the lesson but there has been no passing grade defined for the lesson.
3. It would be advantageous for the Learning Management System to provide API message logging so that a communication history may be recreated for debugging or historical purposes. Consideration should be given to the following:
    a. Method of storage, whether it is stored in a database or out to a file.
    b. Is logging on all the time or is there a switch to turn logging on or off? (There will be some overhead when logging is switched on, as opposed to being off.)
    c. If logging is to file, is there a log file for each course, each user, or one large SCORM log file?
    d. How is log data purged? When logging is stored in the database, how often is the table purged? Can the data be deleted or does it need to be archived? If logging is to a file, does the file build in size until it is manually renewed or does the logging mechanism have a trigger that automatically generates a new file? (If the file is left to grow too big, it becomes very impractical to open it in a text editor.)

Conclusions

SCORM has a proven track record in the web based e-learning world and is the accepted standard in the industry. Reputable LMS vendors should support your SCORM course out of the box. The SCORM 2004 version has added much more control and sequencing at a more granular level than SCORM v1.2 however those differences are significant enough to cause the user base to be slower in adopting the SCORM 2004 version.

411 Waverley Oaks Road
Building 3, Suite 319
Waltham, MA 02452
www.syberworks.com
1-888-642-7078

About the Author

Stuart Campbell is Director of Software Development for SyberWorks, Inc., a privately-held supplier of e-Learning software and training. A native of the United Kingdom, he had previously served as a Principle Software Engineer, Senior Consultant, Senior Software Engineer, and Development Specialist for companies such as Brooks Automation Inc., Digital Equipment, and Honeywell Control Systems. His areas of expertise include Visual Studio.NET, C#, VB.NET, VB6, VBScript, XML, COBOL, WindowsXP, Windows2000, WindowsNT, VAX/VMS, UNIX, Oracle, SQLServer, Oracle Rdb, Oracle DBMS, and Agile Modeling Methodology.

About SyberWorks

SyberWorks, Inc. (www.syberworks.com) is a leader in the custom e-Learning Solutions and Learning Management System industries for Fortune 1000 corporations, higher education, and other organizations. Located in Waltham, Massachusetts, the company serves the multi-billion-dollar e-Learning market. Since 1995, SyberWorks has developed and delivered unique and economical solutions to create, manage, measure, and improve e-Learning programs at companies and organizations in the United States, Canada, Europe, and other countries.