LearnUpon

Ultimate SCORM Overview

Table of Content

About the ebook

We have written this ebook to provide you with an overview of the key elements involved in SCORM and to endeavour to demystify it. There are several sections to the paper which are clearly detailed in the table of contents to follow. In summary, the key sections include the following:

> ***What is SCORM?***
>
> ***How is SCORM created?***
>
> ***Should you use it?***
>
> ***How can you get a SCORM course into your LMS?***

Your prior knowledge and exposure to eLearning will determine your level of familiarity with the various eLearning terminologies you will come across throughout this ebook. For simplicity, we have included a table of terminologies to act as a guide. It’s by no means an exhaustive list but aims to explain some of the more common terms.

Terminology

AICC: Aviation Industry Computer-Based Training Committee. The first globally recognised eLearning Content Standard developed in the early 90's by a number of leading aircraft manufacturers.

Authoring Tool: A content authoring tool is a software application used to create multimedia content which is delivered through an LMS.

CMI-5: CMI-5 is an xAPI profile. It can be considered xAPi with extra rules and that is more adaptable to modern technology.

LETSI: International Federation for Learning, Education, and Training Systems Interoperability.

LMS: Learning Management System. A standard to connect systems for the administration and delivery of elearning courses.

LTI: Learning Tools Interoperability, to connect systems such as a learning management system (LMS) with external service tools in a standard way across learning systems.

Manifest file: A manifest file in computing is a file containing metadata for a group of accompanying files that are part of a set or coherent unit.

RTE: RTE covers the point from which a course is launched and determines how information, including scores, answers or bookmarks, are tracked back to an LMS.

RTWS: Run-Time Web Service API.

SCORM: Sharable Content Object Reference Model, is a collection of standards and specifications for e-learning.

Tin Can/xApi: A relatively new specification for learning technology that makes it possible to collect data about the wide range of experiences a person has (online and offline).

What is SCORM? The basics

SCORM has become so widely used, it's often referred to as the default or de facto standard of eLearning. But SCORM's confusing name and technical nature can make it difficult for eLearning professionals to understand. Here are the basics you need to know to answer the question: what is SCORM?
The word SCORM is an acronym and stands for: Sharable Content Object Reference Model. So it's not difficult to see why so many people find SCORM confusing! But the two parts of the SCORM acronym are actually pretty easy to understand once we separate them out. The SCORM term is made up of two elements that, when combined, answer our question, "what is SCORM?".

The two main components of 'SCORM' are:

- Sharable Content Object and Reference Model.
- Sharable Content Object: This describes the elements of the SCORM package that can be

reused across multiple tools and platforms. Once the various elements of the package are SCORM compliant, the content should be understood by all compatible learning platforms and tools.

- Reference Model: This part of the term tells you that SCORM is a standard, the specification for which can be understood and applied in a consistent way by all who work in the eLearning industry.

And that's the heart of SCORM: a set of technical specifications that were developed to provide a common approach to how eLearning content is developed and used. Since then, SCORM has also gone through a number of iterations. To check if eLearning content, or a tool or system, really is compliant, ask a vendor or developer to explain the details. For example, is the product compatible with all elements of the SCORM spec?

History of SCORM

Believe it or not, day one of our SCORM series starts in 1999, with the United States government. Back in the day, many US government departments conducted online training. At the time, however, the co-ordination and consistency of training provided was pretty much non-existent. If you can imagine, multiple departments produced reams of course content, duplicated it, generated inconsistent completion and exam results, and so forth. That was life pre-SCORM! The US Department Of Defense (DoD) and Advanced Distributed Learning (ADL) decided to combine their collective experience on the subject, under an initiative issued by President Bill Clinton, SCORM was soon born. It would be wrong, however, to describe SCORM as the first “standard”. AICC (Aviation Industry Computer-Based-Training Committee) deserves a mention too. AICC was, indeed, the first official eLearning content standard. Developed in 1993 as a CD-ROM based standard, online web support was added to its specifications in 1998. However, AICC’s interoperability and ease of use (when imported to an LMS, for example), made it very difficult to work with. Many steps were often required just to get content in the format up and running in an LMS.

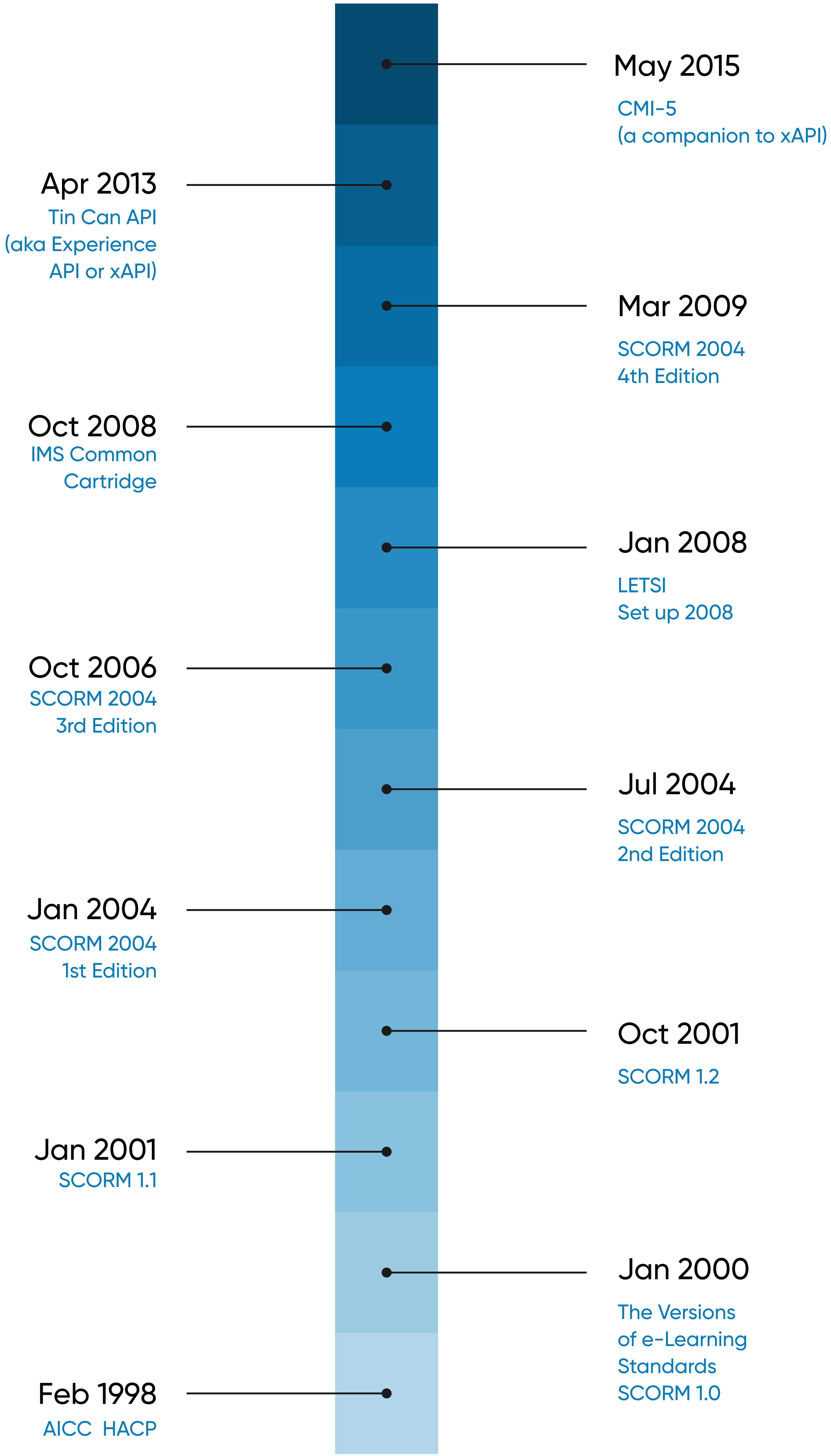
Versions of SCORM

In its infancy, SCORM was released as version 1.1. Undoubtedly, it started a revolution. While the version lacked some key concepts, it's still the ground on which SCORM stands today. In 2001, version 1.2 was released in order to improve on some concepts that 1.1 failed to address, like the lack of metadata, and to offer greatly improved interoperability. Version 1.2 has been a big hit with the industry and is still heavily used today. Arguably, it remains the most popular version in the SCORM family.

That status quo remained until 2004 when (cleverly enough!) SCORM version 2004 was born. Up to 2009, four editions of the 2004 version were released, each refining what preceding versions had delivered.

The most notable advance the 2004 version offered over 1.2, was the ability to sequence content (i.e. to control the order in which learners complete course sections or modules), and an enhanced ability to track how learners navigate between module subsets. SCORM 2004 also introduced a number of new features. The release of new tracking fields became a hotly contested element of the specification. The field for storing learner bookmarking data increased from 4,096 characters to a much-needed 64,000 characters. But funnily enough, the size only increased in the third edition. The second actually reduced it to 4,000 characters, causing an outcry!

But what are the implications of all of these versions in reality? In our view, SCORM 2004 introduced too many editions of the format. Each included too many changes, even reducing the size (if only temporarily) of a bookmarking field that was already a bone of contention in the industry. While the format continued to move forward, it also took a number of steps back, introducing some concepts that confused developers. As a result, the 2004 version ended up being more about what data could be tracked, than a true addition of new features to the specification. Ultimately, version 1.2 has prevailed as the preferred solution and is still more widely adopted. In our opinion, that's as it should be because the version is less ambiguous. In most cases, a sensible LMS implementation will allow a SCORM to track larger bookmark sizes than the 4,096 specified in 1.2. But more on that later – We'll save the real technical gems for later sections.


May 2015
CMI-5
(a companion to xAPI)
Apr 2013
Tin Can API
(aka Experience
API or xAPI)
Mar 2009
SCORM 2004
4th Edition
Oct 2008
IMS Common
Cartridge
Jan 2008
LETSI
Set up 2008
Oct 2006
SCORM 2004
3rd Edition
Jul 2004
SCORM 2004
2nd Edition
Jan 2004
SCORM 2004
1st Edition
Oct 2001
SCORM 1.2
Jan 2001
SCORM 1.1
Jan 2000
The Versions
of e-Learning
Standards
SCORM 1.0
Feb 1998
AICC HACP

To SCORM or not to SCORM

Should you use SCORM?

We work with a lot of companies that are relatively new to eLearning, and one question that comes up a lot is: "Should we build SCORM courses or should we bring our existing content (in Word, PowerPoint, PDF, or video formats) directly into LearnUpon instead?" It's still not a straightforward question to answer. There are many pros and cons that can influence a decision about whether to build SCORM-based courses using a popular authoring tool like Adobe Captivate or Articulate Storyline alone. It depends on your individual circumstances and what you're trying to achieve with the eLearning program you plan to develop. Here's a list of the main pros and cons of SCORM to consider before making a decision.

Let's start with the pros of developing SCORM-based content to import into your LMS:

- Developing courses in SCORM can help you to make eLearning content more interactive. Many SCORM-compliant authoring tools allow you to add engaging features: directing learners to click on images and other onscreen elements, rendering dynamic text as a result, asking learners to expand components before they proceed to the next section, the inclusion of mini-quizzes embedded within the course, to name a few popular tactics. And as most of us working in eLearning believe, the more engaging and interactive your course content, the more likely it is that learners will, firstly, complete it, and, secondly, retain what they've learned after they finish the course.

- A SCORM package will also allow you to control the length of time a learner spends on a course before it's marked as complete. SCORM-compliant authoring tools include settings that force learners to spend a specific amount of time on each page of a course before the "Next" button is enabled. These features allow you to design a course of a minimum length, for example 50 minutes, which is required for some accreditation bodies in a number of States. While this setting doesn't mean the learner is more likely to pay attention to your content while they wait for the "Next" button to enable, it does allow you to meet requirements that stipulate that learners must spend a minimum length of time on the course.

- A SCORM package can also allow you to combine course and assessment content to deliver a superior user experience. If your course must be assessed with a final exam, for example, an authoring tool will allow you to place it directly after your course content and return the score to your LMS. It's a popular option with some LearnUpon customers because it means the whole course (content and exam) are contained in a single SCORM unit. That makes life easier for learners who don't need to navigate away to a separate exam once they've completed the course.
- A final advantage of choosing SCORM is that, in theory at least, it should be easier to migrate course content if you need to move to a new LMS vendor. The point of the SCORM standard is that courses built in the format should run in any LMS that's SCORM compliant. We say "should" for a reason. From our experience, there are learning management systems that claim to be SCORM

compliant, when in reality they struggle to run SCORM courses effectively.

- There are also a number of cons to choosing SCORM that you should be aware of before reaching a decision:
- Authoring tools can be quite expensive, particularly the good ones. The two most popular authoring tools that publish content in the SCORM format are Adobe Captivate and Articulate Storyline. Both tools are relatively expensive so if 2 or 3 people in your company need to create courses, a SCORM-compliant authoring tool can be a significant investment. There are some cloud-based authoring tools on the market now, such as Elucidat, which allow multiple instructional designers to work on a course at the same time. These tools are certainly worth considering if you have a team of people designing and developing courses.
- One drawback to consider is that authoring tools take a while to get the hang of. While basic courses are easy to create, you'll probably have chosen the

SCORM format because you want to develop slick content with lots of interactions. But learning how to create sophisticated course content with an authoring tool can take time. You might even decide to invest in a training course to speed up the process, which will add another cost in addition to the license fee you'll already have paid. Here are two guides our Customer Support team has written on publishing SCORM content in Articulate Storyline and Adobe Captivate.

- The most effective SCORM content has traditionally been developed in and published to Flash. If most of your learners will be using Apple iOS devices to access courses, Flash-based SCORM content can cause headaches. While most popular authoring tools now tackle the problem by providing a HTML5 output option, those outputs can be buggy and of inferior quality. If your content contains a lot of rich media, like video, HTML5 can deliver a diminished user experience. And video-based SCORM content can struggle to stream or buffer properly in a HTML5 player.

- The SCORM standard has been used in eLearning for quite a while. While we don’t expect it to disappear anytime soon, in the last few years the Tin Can (xAPI) has challenged its dominance. Before deciding to create courses in SCORM, it’s worth investigating if Tin Can may be a better option. Consider whether investing in SCORM now is likely to make sense in a few years, with the future of the format less than certain. Many of our customers have recently opted for Tin Can, due to the advantages it offers for mobile learning, an area where SCORM can struggle. The most popular authoring tools now also support Tin Can. So if you choose to go with SCORM now, you’ll still be able to convert courses to the Tin Can format in future if necessary.
- Because SCORM is a relatively old eLearning standard, it can be clunky and complicated in places, particularly in the area of course completion, and the communication of statuses back to your LMS. We recommend you select an LMS that provide support to assist customers who

have experienced these kinds of issues with SCORM. Remember not all LMS vendors offer the same level of support, making it tricky for some organizations to get SCORM courses tracking properly in their LMS.

Deciding whether to invest in SCORM courses isn't straightforward and there are pros and cons to both options. If you have time to invest in developing nice content, SCORM courses are a great idea, as they tend to look polished and engaging.

Much will also depend on the type of training you want to deliver and why you need to deliver it. If you're a professional training company that wants to sell courses online, we recommend investing in the the development of professional-looking SCORM-based courses. If you expect someone to pay $50-100 or more for a course, you want them to see and appreciate its quality. If, on the other hand, you simply need to deliver internal training that's less about learning and is more compliance focused, than adopting a simpler approach of importing PowerPoint and video documents with an exam bolted on at the end may be a more cost-effective option.

How to use SCORM compliant authoring tools

The purpose of SCORM is to increase ‘interoperability’. By providing a standard of communication that can be applied consistently, SCORM makes it easier for eLearning professionals to share and migrate course content across tools and systems. How is a SCORM compliant course created?

SCORM specifies how tools and content communicate

Many eLearning professionals mistakenly believe that SCORM somehow defines the nature of an online course. While that's incorrect, the difference is subtle. SCORM is a 'specification', essentially a communications protocol. As an analogy, think about truck drivers (yes, Jerry Reed, Smokey and the Bandit stuff!). A driver follows a specific protocol to communicate with other drivers. Before making a statement or asking a question, the driver introduces him or herself with "Breaker, breaker" and ends with "Over", to signal that they've finished speaking. Another driver might respond: "That's a 10-4", to confirm the statement and end with "Over" to signal that they've also finished speaking.

That's a simple example of a rudimentary protocol for CB radio communications between drivers. We can develop the analogy to map it to SCORM:

- SCORM can be compared to the protocol of using terms like "10-4", "Over" and "Roger" to confirm or signal the end of communications between drivers.

Like SCORM, the protocol helps drivers to communicate by providing a shared set of rules that are consistently applied to minimize confusion.

- SCORM doesn't care what kind of truck you're driving (or LMS you use) or which brand of radio (or authoring tool) you prefer. The role of a protocol like SCORM is to help drivers to communicate with each other. In theory, they could be using a three-wheel truck, milk-cart, or pair of rollerblades. Once they know how to talk CB, they can communicate, share information, and be 'interoperable', just like SCORM.

So, SCORM is a specification that provides rules for how things communicate with each other. It doesn't influence the color, size or complexity of a course. But it does define how the course talks to an LMS – just as truck drivers all over the world use a protocol to communicate, regardless of which vehicles or radios they prefer. And just as CB chatter evolves across time and territories, SCORM has also changed since the first version was published in 1999. Communication protocols always evolve. Next, we'll explain how your vehicle is assembled, that is, how a SCORM compliant course is created, and what it contains.

SCORM compliant authoring tools

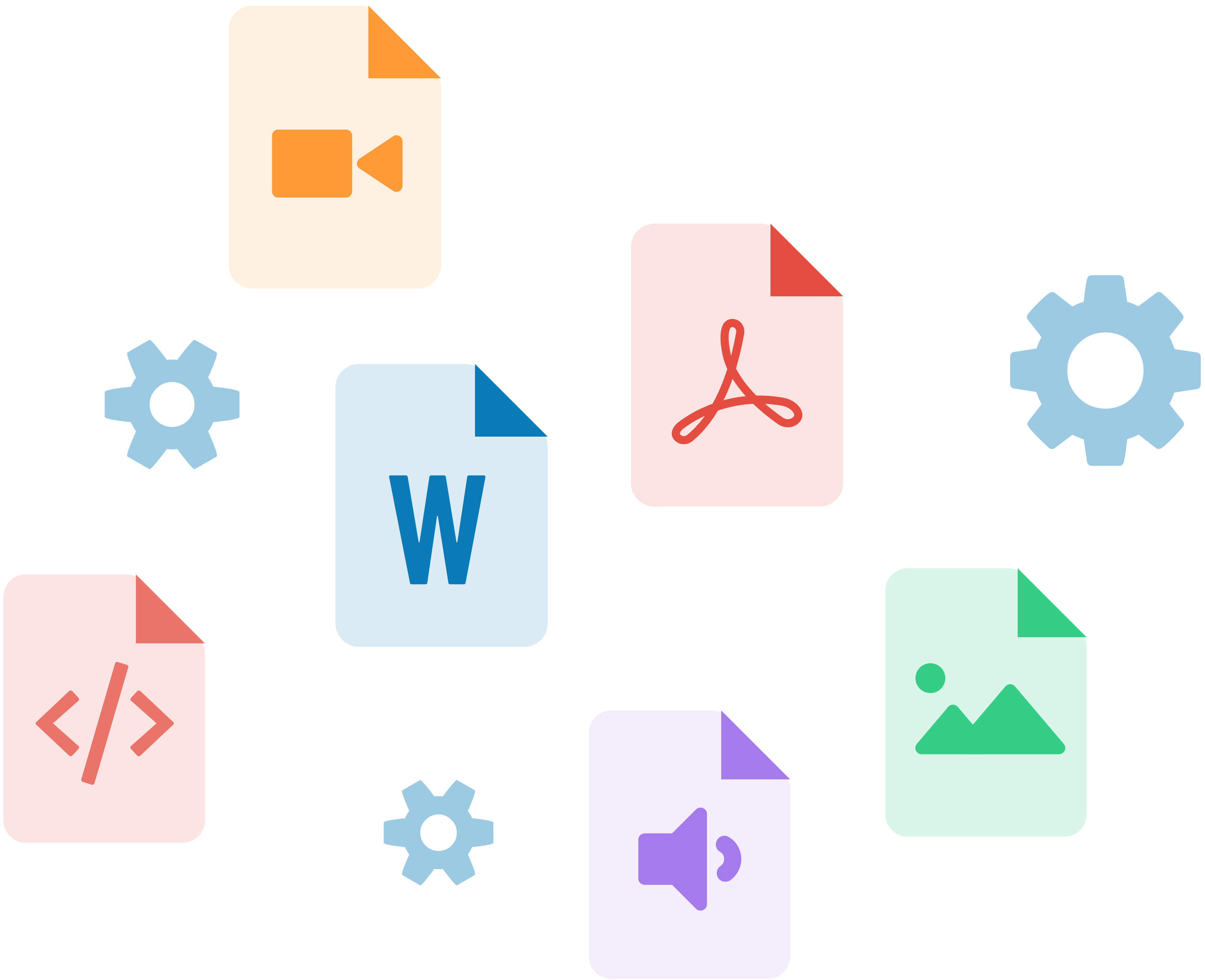
There are two main options for creating a SCORM compliant course. Either you can use an off-the-shelf authoring tool or you can build a tool yourself from the ground up. The latter requires a deep understanding of the SCORM protocol, as well as development skills. Building a SCORM compliant course gives you full control over how a course is developed and the details of learner experience. Navigation and design choices aren't limited by the features of a generic authoring tool. Courses developed in this way also tend to work better, as they can be designed to suit the organization's specific needs and target the individual browsers and devices they support. With that said, an off-the-shelf authoring tool makes course development accessible, particularly if you lack necessary skills in-house.

A wide range of SCORM compliant authoring tools are available, from the simple to the highly complex. Usually, an authoring tool's workspace looks something like PowerPoint. That's a loose generalization but it's fairly accurate. Most authoring tools allow you to import media, build a slide set, and define rules about how learners navigate between slides or pages. That covers the basics of what a SCORM compliant authoring tool must deliver: a course composed of multiple pages or screens that a learner clicks through from beginning to end. A more sophisticated authoring tool will allow you to add interactive elements, like exams, surveys, and rich media including audio and video. The most advanced authoring tools will allow you to create nice transitions between screens and set rules about how learners progress through a course.

Which authoring tool you select should be determined by your requirements, skills, budget and time available to build a course. Before using an authoring tool, sketch an outline of the course to help define your requirements. Some authoring tools prominent include: Elucidat, Articulate Storyline, Adobe Captivate, iSpring Pro, and Camtasia. You can use any of these tools (listed in no particular order of preference) to create SCORM course content. Each has particular strengths. Some, like Camtasia, specialize in the creation of video content. Captivate and Storyline are particularly suited to adapting existing PowerPoint content. While Elucidat specializes in easy cloud collaboration for the development of fully responsive content.

Creating a SCORM package

Once your course has been created, you're ready to export what's called a 'SCORM package'. Remember, SCORM is essentially a communications protocol. Your



completed course might contain imagery, text, video, and a quiz. But that doesn't make it a 'SCORM course'. In order to SCORM-ify the course, you'll use the authoring tool's export option. To use an analogy, when you create a Word document, you can save it to formats like PDF or RTF. When you use an authoring tool, you can also export content to multiple formats, including Tin Can (xAPI), web format (which provides a folder you can post on a website to host a course), or, indeed, SCORM.

If you select the export to SCORM option, your authoring tool will use all the elements of your course (including text, images, video, navigation settings, quizzes) to create a SCORM package. That package gives you two important things. Firstly, you can reuse the package by importing it to an LMS that supports SCORM. That's what makes SCORM 'shareable'. And secondly, the package understands the SCORM protocol, the API specification itself is embedded in your course by the authoring tool, delivering all the interoperability of SCORM.

An authoring tool may also provide options for exporting to a specific version of SCORM, like 1.2 or 2004. But whichever version you choose, you'll achieve the main goal of exporting your course to a SCORM package. The package is effectively a .zip file. The file contains all of your media (the course's images, videos, buttons, quizzes, etc.), as well as some "magic dust" that allows your chosen LMS to import, read, and ultimately to run the course, delivering it to the masses.

You now understand that SCORM is a specification or a protocol. You still need to build your course in the usual way and to create a SCORM package with its contents. SCORM won't make the content, usability, or structure of your course better or worse. That's up to your instructional designer or course developer. But SCORM will allow you to share your course, run it on an LMS, and track learner progress and results.

Getting SCORM content into your LMS

At this point, you understand how to create a great SCORM compliant course with an authoring tool, and we have discussed how to export the course as a SCORM package. So we've now SCORM-ified our hard work and the content is ready to be delivered to learners.

When you export your course to a SCORM format, two important things happen:

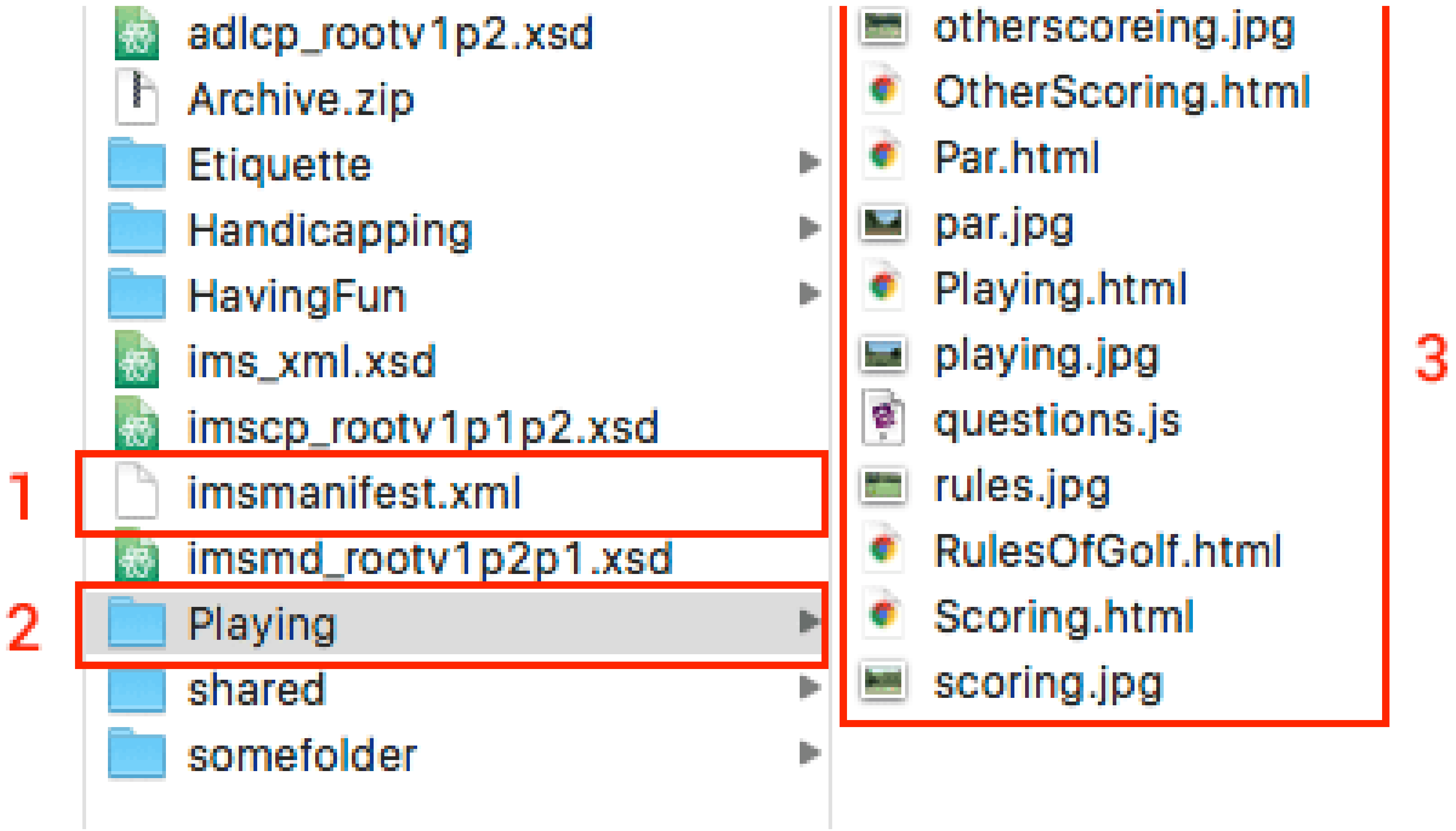
1. Your authoring tool creates a SCORM zip file. That zip contains all of the media files that enable learners to view and navigate your course online, including text, imagery, video, CSS, and quiz content. If your authoring tool has done its job correctly, the contents of the zip will be neatly separated into folders and files that reflect the content of your course.

2. While you now have a zip file, two things in particular define it as a 'SCORM' zip:

- Inside the zip is a file called imsmanifest.xml. This manifest file specifies the contents of the zip. For example, an aircraft's manifest file would detail the names of passengers, where they're seated in the aircraft, the weight of the cargo in the hold, and a wealth of other information about the aircraft and its "contents". The imsmanifest.xml serves a similar purpose – it details the contents of the zip package.
- Your authoring tool will also have included files that it didn't create but that are SCORM specific. These are called the SCORM API communications and logic files and are at the heart of why the package can be defined as a SCORM course. The files allow your course to communicate with a SCORM compliant LMS that understands the same SCORM protocols.

On the following page there is a screenshot of typical zip contents that have been opened for viewing:

Three takeaways from this screenshot:



1. The screenshot shows the imsmanifest.xml file sitting in the root of the SCORM package zip.

2. This is an accurate but simple example of the folders and structure of a SCORM zip. Complex courses could contain many folders and additional folders within this hierarchy.

3. In this simple example of a SCORM course, the screenshot shows the contents of a “Playing” folder – marked 2 in the screenshot. As you can see, there’s nothing particularly ‘SCORM’ about it. It’s just a folder containing html, images, and javascript files, which means it contains online content that can play in a

browser to be viewed by learners. As those with experience of SCORM will know, the “shared” folder contains all the SCORM API magic that allows the course to communicate with an LMS when it’s launched. The imsmanifest.xml coupled with that “SCORM API magic” is what makes this zip a SCORM package.

The Manifest

Remember that the zip file contains an imsmanifest.xml file. One important point to note is that the manifest must exist in the root of the zip file, and nowhere else! When your LMS tries to import SCORM content, it will look for the manifest in the root of the zip file. If the manifest isn’t there, the file goes against SCORM specifications. That can cause a lot of confusion for SCORM creators as an LMS won’t import the zip files.

As the imsmanifest.xml name suggests, the file is in XML format. The XML file will look something like the example below. The file contains key information for your LMS to interpret. While we’ll discuss some of these now, it’s worth noting that this is a basic example. It contains just a single module and enough sample files to illustrate our discussion points. In reality, a course could contain hundreds of entries in the manifest.

```
<?xml version="1.0" standalone="no" ?>
<manifest identifier="com.learning.scorm.series.day.3" version="1">
  <metadata>
    <schema>ADL SCORM</schema>  1
    <schemaversion>1.2</schemaversion>  2
  </metadata>
  <organizations default="learnupon_default_org">
    <organization identifier="learnupon_default_org">
      <title>LearnUpon - SCORM Series Day 3</title>
      <item identifier="lup" identifierref="launch_page">  3
        <title>LearnUpon - SCORM Series Day 3</title>  4
        <adlcp:masteryscore>85</adlcp:masteryscore>  5
      </item>
    </organization>
  </organizations>
  <resources>
    <resource identifier="launch_page" type="webcontent" adlcp:scormtype="sco" href="module1/start.html">  6
      <file href="module1/start.html"/>
      <file href="module1/logo.jpg"/>
      .
      .
      .
      <file href="module1/style.css"/>
    </resource>
  </resources>
</manifest>
```

Six takeaways about the manifest:

1. Schema: The manifest should contain this information to record which “SCORM schema” is being used. This will typically be ADL SCORM.

2. Schemaversion: This tells us which version of the SCORM specification the course is compliant with. In this example, the relevant version is 1.2 but the schemaversion could also specify 2004 2nd Edition, or 2004 3rd Edition, etc.

3. Items: These are effectively the modules for your SCORM. While SCORMs can contain multiple modules, one is most common. In the example, we've drawn an arrow from point 3 to point 6. That's to illustrate that the indentifierref is important. In this case, "launch_page" links to a resource. In other words, the item has a resource, which it's linked to by the id launch_page. That resource defines what type of item it is and where the content for the module is located. We'll describe that in detail a little later.

4. Title: This defines the title of the module. In our example, that's "LearnUpon – Getting to grips with Scorm". In real world examples, you'll often encounter multiple items, each with their own titles.

5. Masteryscore: This element defines a passing score for the module. If your module includes a quiz, you may have set a passing score – but not every SCORM course includes a quiz and not every quiz has a passing score. Knowledge checks and practice quizzes typically won't have a masteryscore, for example.

6. Resource: One important type of resource (as explained in point 3) is a "sco", defined by the attributeadlcp: scormtype. A "sco" is what's called a launchable resource. That means it's a module learners can launch, view and complete in a browser. You'll note that this resource has an href attribute, of "module1/start.html". That tells your LMS – "When a learner launches this module, open the page module1/start.html for them". In other words, it's how the starting page of your module is defined.

What could possibly go wrong?!

While a manifest file looks simple once you break it down into its various parts, things do go wrong. Common issues include:

1. Some authoring tools don't "XML Escape" files correctly. If your course title contains a symbol like "&",

that can cause breakages in some authoring tools, especially in older versions.

2. The manifest doesn't state the SCORM version. For example, the manifest might specify a SCORM 2004 course, while your LMS reads the content as a 1.2. Or the manifest could define a SCORM 1.2 course and your LMS might not support that version of the specification. That's why it's essential to ask your LMS provider which versions of the SCORM specification they support before exporting course content to a SCORM package!

3. If your course has no adlcp:scormtype defined, then your LMS can't determine which page to launch, as your course has no starting point.

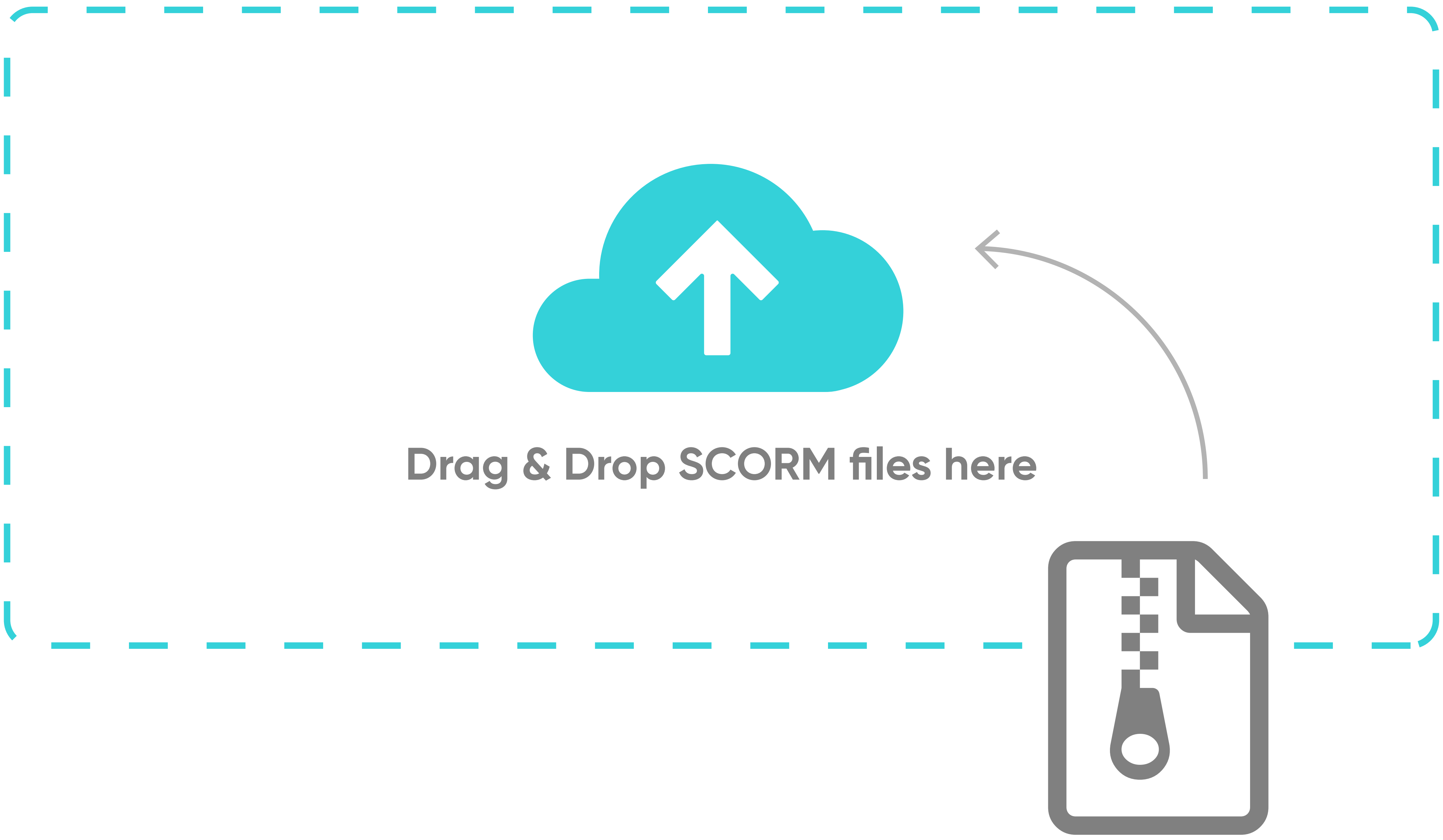
4. If your course includes a quiz but has no masteryscore defined, then the pass/fail determination can also break.

5. One serious issue could be that your zip file doesn't have a manifest. Or the manifest may be located

somewhere outside of the root, possibly embedded deep in the zip file or located in a folder a level deeper into the zip.

Importing SCORM content to an LMS

While importing SCORM content can be a laborious process, it should be simple. To import a SCORM course into your LMS, you will usually:



1. Upload the SCORM zip file, via your browser or LMS administrator UI. For example, to add a SCORM module to a course in LearnUpon, you simply click a button.

2. Your LMS will then attempt to find the manifest file and import its contents, preparing it for use by learners. A good LMS will attempt to validate the manifest and flag potential issues. For example, if no manifest file was found, the LMS should notify you of the specific issue rather than saying something vague like: “Failed to upload your file, please contact your administrator”!

We hope you have a better understanding of what a SCORM package actually looks like at this point. We will now look at how the information is actually relayed to your LMS.

Getting to grips with SCORM API

The magic of the SCORM API enables learners to launch or resume a course within the LMS and progress and results to be tracked throughout.

Key runtime environment specs

The runtime environment (RTE) is what we’re interested in today. RTE covers the point from which a course is launched and determines how information, including scores, answers or bookmarks, are tracked back to an LMS. The RTE specification is large and, at first glance, quite daunting! However, we'll break the flow into basic sections and rules to make the specification easier to digest. Let's jump straight in! When a learner launches a SCORM course in an LMS, the following "actions" kick into play:

1. **"FindAPI":** Let's presume that a learner has clicked a button on the LMS UI to launch a course. A SCORM will first attempt to find the API provided by an LMS in order to track data. Say what?! Yep, that's right. An LMS provides an API for the SCORM to use. That might sound complicated but it's not really. The API is essentially a simple JavaScript object that presents all SCORM RTE functions for use. For example, a SCORM course will search for the API object, after which it will be able to call "save", "get", "start" and "exit" type commands. So the API object is a piece of JavaScript that manages communications between the SCORM and the LMS. But more on that later! The object, however, doesn't have to be JavaScript-based. It could easily be created using

another language, like Java or Flash. But given the scrutiny both technologies are under, and their love/hate relationship with firewalls, JavaScript APIs represent a sensible approach.

2. **LMSInitialise:** This piece of SCORM fancy talk allows the course to say: “Hey, a learner started the course, or learners launched the course, so you, LMS, do stuff!” As in step one, the API object was found and now the SCORM is calling its LMSInitialise function. That tells the LMS that the SCORM is starting up and it should prepare course data for use.

3. **LMSGetValue & LMSSetValue:** If the initialization succeeded (and more on that below), the SCORM may now call some methods on the API to get or set data. For example, the SCORM might “get” information about the last bookmark a learner accessed from the LMS. When a command like that is received, the LMS responds by sending bookmark data for the course to SCORM, so the course can resume where the learner last exited. The set describes the opposite scenario,

funnily enough. Set can be used to tell the LMS which bookmarks it should store for the learner. In other words, set says: “Hey you, LMS, the learner is on page three now, store that!” The LMS obeys and returns a “success” message to the SCORM course. The SCORM now knows that the bookmark was saved and it doesn’t need to get angry about the potential for lost data. Ultimately, get and set are used to retrieve and save data, not only for bookmarking, but also statuses, scores, and so forth.

4. **LMSFinish:** The opposite to LMSInitialise, LMSFinish is called when a learner exits a SCORM course. It doesn’t tell the LMS that a learner has completed the course, just that they’ve exited. There are many different steps that can occur at this point. If, for example, the learner has actually completed a course or passed an exam module, the LMS will need to act on that – we’ll cover some of those details below. For now, it’s important to understand that, as far as the process of a learner launching, viewing or completing a course is concerned, the SCORM protocol and specification can be boiled down to the four steps outlined in this section.

Key SCORM API attributes

As we've discussed, the LMSGet/SetValue commands are essential for a SCORM to track data back to an LMS. This Ebook would be too long and sleep-inducing if we had to cover all of the possible attributes. But there are some key concepts that are worth summarizing, so that you're armed with enough knowledge to feel comfortable with common SCORM lingo.

SCORM RTE defines some attributes or fields that are specific to the SCORM data realm. For example, an attribute for storing scores is defined as score_raw and is a form of numerical field. Here, we'll discuss the most common attributes and uses:

- Read/Write and sizes/types: The SCORM API includes many attributes that can be updated and read from. For example, there are attributes for scores, statuses, bookmarks, quiz answers (which are called interactions), and more. Each attribute has its own type (be they numbers or boolean true/false, etc.). They also have a definition that states whether or not the attribute is read only or read/write. A good example of this occurs when a SCORM calls an LMSGetValue to retrieve a learner's name (which is read only because a SCORM can't change names in an LMS). The flipside is that if the SCORM tried to call LMSSetValue for a learner's name, the LMS should give back an error. Indeed, if the value is a writable field, the LMS should be storing it for later use (probably in its own datastore).

- suspend_data or lesson_location: These attributes are typically used for bookmarking data and are examples of a read/write type field. For example, a SCORM might use lesson_location to set a number like 12, indicating that the learner is on page 12 of the course. That information is stored by the LMS to use when the learner launches the SCORM again. While suspend_data is similar, it also supports larger data volumes: 1024 characters, to be exact (which was extended to 64,000 in version SCORM 2004 3rd ed). suspend_data makes it possible to store larger kinds of bookmarking data, like pages visited, but also answers submitted to questions, or bookmarks in quizzes and subsections of pages. As an example, suspend_data could look like: 1,3,4,5,10|answered_1,wrong|answered_2|wrong. The data here is "random", in that it's not SCORM specific, but specific to the course or authoring tool that set it.

- lesson_status: This describes the status of a learner's course. lesson_status will at first be "not attempted" and the SCORM will later set a status as appropriate. For example, if a learner exits half way through a course, you'll typically find the status set to "incomplete". Similarly, if a learner completed their module or passed or failed the course, the SCORM will use lesson_status to tell the LMS that the learner's status is "completed", "passed", or "failed". It's worth mentioning one pro-tip that confuses SCORM creators and causes tracking issues. If the SCORM doesn't set lesson_status (so if, from start to finish, no value was ever set), the LMS can assume that the learner completed the course on exit!

- session_time & total_time: These fields are extremely useful for eLearning reporting. They allow you to track the amount of time a learner takes to complete a module and how much time they spend on a course overall. Session time is a writable field that allows a course to tell an LMS how much time the learner spends viewing it during a given launch and exit session. The LMS will take the session_time value and tally it up into a total time tracked. That way, the LMS can track total time across multiple launches if necessary. The SCORM specification views total_time as a read only field. That means the LMS, and not the SCORM, is responsible for managing the data.
- score_raw & mastery_score: we mention these together as they usually go hand in hand. score_raw is the value used to store a mark calculated for a course, for example a learner might achieve 85% or 33.22%. The mastery_score, as we learned previously, is the score needed to pass a module. That's very important to the LMS, for one reason, the LMS can determine if the learner passes or fails (i.e.

can change a completed status to passed or failed) when the following holds true:

- The learner has scored something, i.e. score_raw has a value.
- The SCORM has set lesson_status to “completed”. It’s important to note that completed is different to passed. You might complete reading a book but that doesn’t mean you passed or failed! It’s possible to just complete some courses and pass or fail others, so the difference is subtle but important.
- The SCORM has a mastery_score associated with it.
- If these three steps hold true, the LMS can compare a learner’s score to the mastery_score for the course and determine a pass/fail status. If the learner’s score is greater than or equal to the mastery_score, the learner will be deemed to have passed and the LMS will set that status. And if the learner didn’t achieve the necessary score, a fail would be recorded. In this scenario, the LMS can’t change the status itself. It must listen to and obey the data the SCORM shares without altering it. Otherwise an LMS

would be said to overwrite history, which is prohibited by the SCORM standard, and rightly so.

- Interactions/Objectives: SCORM defines attributes for tracking user objectives and interactions within a given module. You can think of objectives as key goals a learner must reach, and interactions as the answers a learner submits during a quiz or survey in the SCORM. It's particularly important that the specification defines useful elements the SCORM can query the LMS for, such as how many interactions were saved. The SCORM can also store, not only answers submitted, but the amount of time taken to submit an answer, the correctness of the answer, and scores achieved. That data can be very useful for creating reports in your LMS. While knowing which answers were submitted during an exam is great, knowing the length of time taken between answers allows you to analyze some pretty neat metrics later.

Mandatory vs. non-mandatory

It's worth mentioning that SCORM defines a set of mandatory fields that an LMS must implement in order to be compliant. The RTE specification can be broken down into mandatory and non-mandatory fields. It's important to ask an LMS vendor you're considering if they support all, or just a subset, of the SCORM specification. For example, because quiz interactions are not mandatory, you might find that your quiz won't track answers submitted by learners in your LMS, leaving a serious gap in your reporting capabilities.

Cross-domain issues

It wouldn't be right to finish without mentioning cross-domain issues and SCORM! SCORMs typically track using a JavaScript API object exposed by an LMS. The LMS is required to call this object "API" and it must reside on the window of the LMS launching the SCORM. Now the fun really starts if the LMS is on one domain and the SCORM that's being launched is on another – your LMS may be hosted on mylms.somewhere.com and your course hosted on mycourse.overthere.com. By definition, JavaScript can't communicate easily across domains. While it's possible with the help of some JavaScript trickery, most authoring tools don't provide SCORMs that are cross-domain capable. It's important to ask an LMS provider if they've gone the extra mile to support cross-domain tracking with SCORM. It's particularly important if your SCORMs are media rich and require a Content Delivery Network (CDN, such as CloudFront) to deploy courses. At LearnUpon, we've taken that approach. It delivers a better experience to learners viewing courses that contain rich-media content delivered via the server closest to them, rather than relying on the server that hosts your LMS (which could be far flung!).

Summary

All in all, SCORM has travelled a rocky road since its inception. Like any form of online standard, it took time to mature and settle down on the path to widespread adoption. If you're thinking about using SCORM for eLearning content or are interested in developing a SCORM-based course, we hope this SCORM overview will prove helpful. SCORM is now in competition with new standards, quickly evolving specifications like Tin Can (xAPI), CMI-5, LTI, and CC. But we'll leave those rivals to SCORM for a different Ebook.

LearnUpon

Learning as it should be

About LearnUpon

LearnUpon LMS helps businesses train their employees, partners, and customers. By combining industry-leading capabilities, unmatched ease of use, and unrivaled customer support, organizations can manage, track, and achieve their diverse learning goals—all through a single, powerful solution. It's learning as it should be.

Learn more