ULAM Headless Course Format

ver. 50b2a4b

Mateusz Wojczal. November 2021

Contents

The Abstract

Current e-learning does have a huge technological dept and do not responding to market needs as fast as other segments. The main reason is the obsolete formats like SCORM that are widely used which does not separate data layer from presentation one. There is a need from market of existence of better formats.

The Introduction

Current e-learning formats does not separate data from presentation layers. Current e-learning content are not portable and are not designed to age well.

With separation of layers the content can be displayed in modern way everything some new devices is used. If SCORM courses where designed in this fashion back in 2000s it would be straigh forward to convert them to any devices, like mobile phones, smartwatches, smart tvs, etc. Because of wrong design decision we're stuck with this format and obsolete courses.

Back in the days when Advanced Distributed Learning was creating SCORM adapting older AICC HACP desktop format most of the personal computers used the same browser, on the same operation system with common 1024x768 pixel resolution. If there were variation to this statement they were minimal. Browser were not able to do much more then to show server response in HTML format after client request. Everything showed in browser window was rendered by server and even if there was separation of layers it happened only on server side.

Organizations that are working on e-learning standards are responding to market needs very slowly. Their latest specification `cmi5`, which does solve many of the issues, is already 6 years old and not commonly adapted - the most popular format SCORM 2004 4th Edition was published in 2009.

The headless approach seems to be solving all of the issues that modern e-learning and LMSes do have. The separation of content and it's players allows to create courses that works well on any device and do age well. Course designed in this favour most likely will be able to be played on device not yet used.

Evolution of e-learning

History of e-learning formats

The most popular e-learning formats are created and managed by the Advanced Distributed Learning (ADL) Initiative from the Office of the United States Secretary of Defense.

Before e-learning was used in the web browser environment there was AICC's format created in 1993. First widely used format was AICC HACP released in 1998 which later evolved into SCORM 1.0 that was released in year 2000.

SCORM which is an abbreviation of Sharable Content Object Reference Model since this day is the most popular e-learning package standard. Since version 1.0 to latest SCORM 2004 4th Edition this format is a collection of standards and specifications for web-based e-learning. The format itself describes communications between client side content and a host system and how to package whole course into ZIP files that are called "Package Interchange Format."[1]. Latter is a ZIP package that contains HTML files and XML manifest.

Since SCORM introduced many issues The Experience API, also known as Tin Can API or xAPI was released and later cmi5 format that provides a set of rules intended to achieve interoperability in a traditional Learning Management System environment.

xAPI specification removes content for it description, and allows the content to send "statements" based around [actor] [verb] [object], or "I – did – this" to a Learning Record Store (LRS) which can be part of Learning Management System but can live on their own or as part of another system.

[1]Technical Specification 4th Ed.. SCORM. Retrieved 2017-05-22.

The table below [2] summarizes the comparison of each standard:

Format	Released	Pages	Widely Used	Run-Time	Pack-aging	Meta-data	Sequen-cing	Works Cross Domain
AICC HACP	Feb 1998	337	Yes	Yes	Yes	No	No	Yes
SCORM 1.0	Jan 2000	219	No	Yes	Yes	Yes	No	No
SCORM 1.1	Jan 2001	233	No	Yes	Yes	Yes	No	No
SCORM 1.2	Oct 2001	524	Yes	Yes	Yes	Yes	No	No
SCORM 2004 "1st Edition"	Jan 2004	1,027	No	Yes	Yes	Yes	Yes	No
SCORM 2004 2nd Edition	Jul 2004	1,219	Yes	Yes	Yes	Yes	Yes	No
SCORM 2004 3rd Edition	Oct 2006	1137	Yes	Yes	Yes	Yes	Yes	No
SCORM 2004 4th Edition	Mar 2009	1162	Yes	Yes	Yes	Yes	Yes	No
IMS Common Cartridge	Oct 2008	135	No	No	Yes	Yes	No	Yes
IMS LTI	May 2010	25	In Academic LMSs	Yes	No	No	No	Yes
The Experience API (xAPI)	April 26, 2013	85	Not Yet	Yes	Partial	No	No	Yes
cmi5 (a companion to xAPI)	June 1, 2016	48	Not Yet	Yes	Yes	No	No	Yes

What is Learning Management System - LMS

Web accessible application that takes care of administration, documentation, tracking, reporting, automation, and delivery of educational courses, training programs, or learning and development programs is called Learning Management System. LMS systems are kind of software that manage e-Learning.

The most popular LMS is Moodle [3], released on 20 August 2002 because it's available for free as open course software, distributed under the GNU General Public License.

Moodle is program written in PHP that is being served by machine that use PHP. That means that all of the actions for administrators, course creators, students and any other roles does require to connect to machine (server) that serves Moodle. This is a monolith architecture, which means that all moodle components are PHP based working on one machine that parses moodle source code every time there is a request from the browser. Components of the program are interconnected and interdependent in a tightly-coupled architecture.

Most other popular LMS works very similar, as they monolith architecture is the most popular among the LMS

[2] A timeline and description of the eLearning standards.. SCORM

[3] Moodle - Open-source learning platform | Moodle.org

LMS Monolith Architecture

It the diagram below there is Moodle monolith architecture

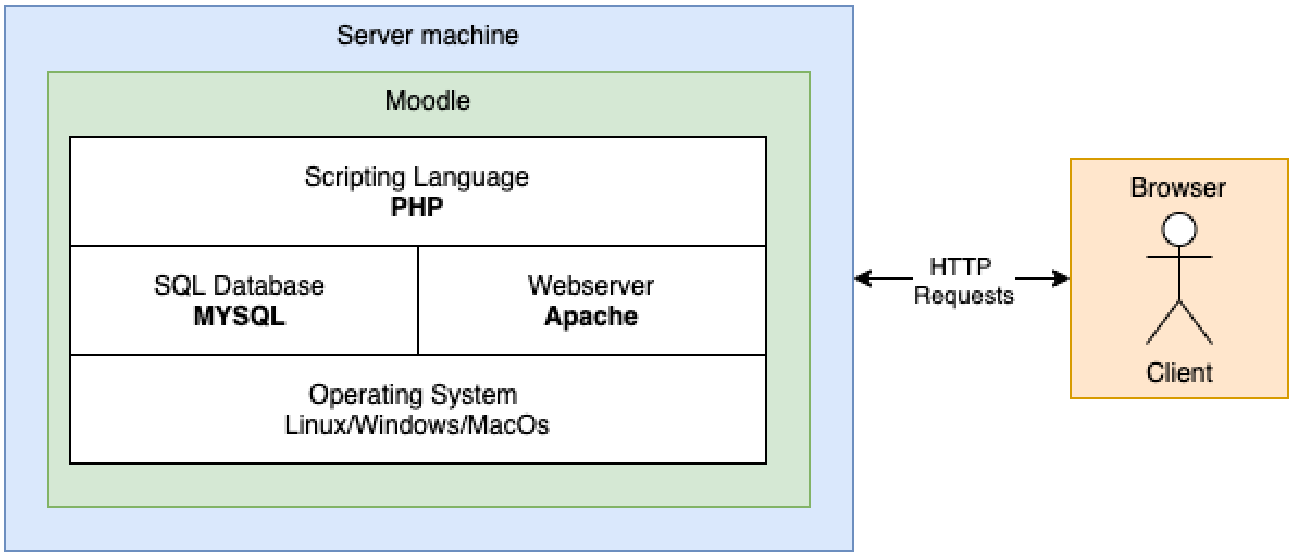

Figure 1: LMS monolith architecture. Moodle technical architecture

All the LMS Features that includes

- Managing courses, users and roles
- Online assessment
- User feedback
- Synchronous and Asynchronous Learning
- Learning Analytics

are handled directly from the server, the response is prepared before being sent in HTML format by PHP preprocessor, the client gets the HTML already rendered document.

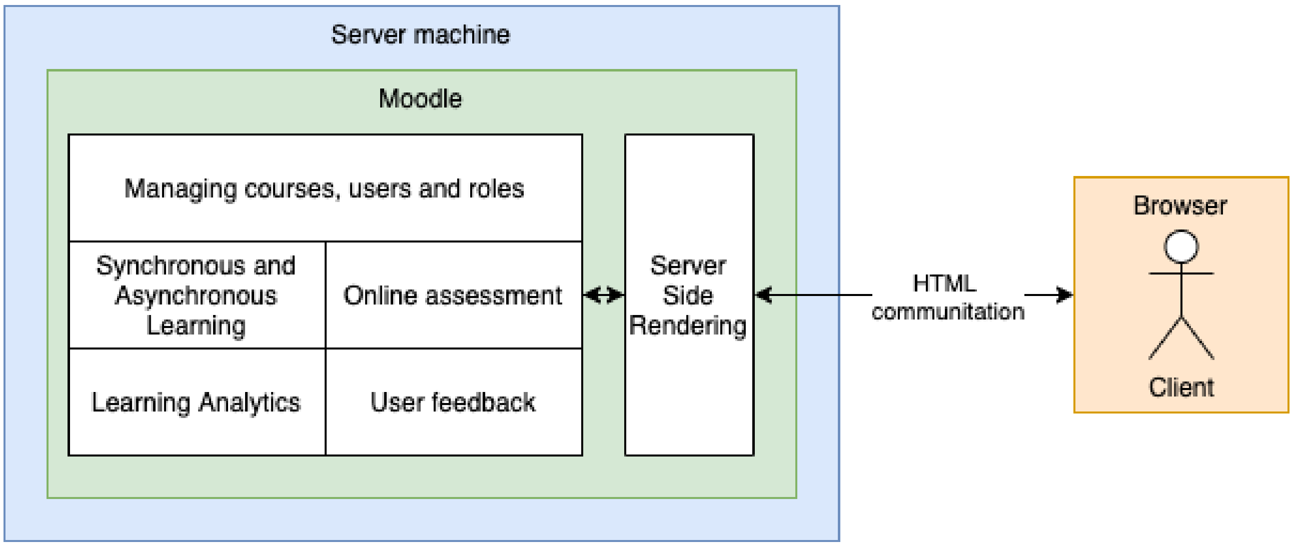

Figure 2: LMS monolith architecture. Moodle functional architecture

All the above means that Moodle and dedicated server is required all the time for all e-learning activities.

Process of publishing the course

Standard way of creating and publishing SCORM compliant course is to follow the steps

1. Creating of a course in an e-Learning authoring tool (like Adobe Captivate [4]) or in from the LMS environment.
2. Course is published as a SCORM package, a ZIP file
3. SCORM package is being uploaded with LMS upload form and prepared to be published
4. LMS publish the course to the students. All results of activities are stored in the LMS

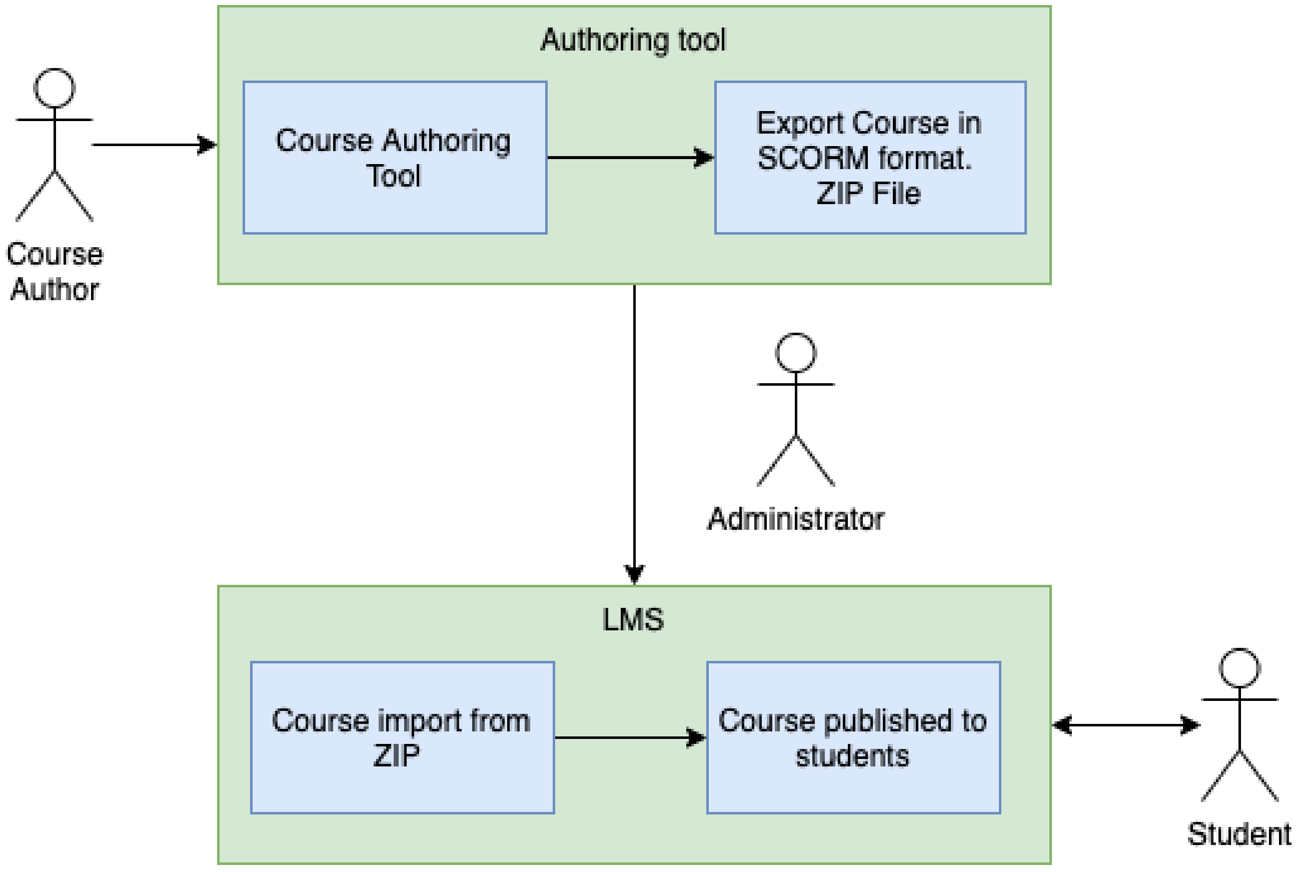

Figure 3: Process of publishing the course

The process above is one direction - which means that SCORM package is closed format, once published it cannot be changed. In order to make any changes, event amending simple typo, the whole process must be repeated - course needs to be changed in authoring tool, then uploaded, etc.

Introduction of Experience API (xAPI) and related technologies

One of the limitation of SCORM that decided about introducing extended formats was capability to track and trace activities from students only within the same LMS. That means that the course and LMS are inseparable.

xAPI specification removes content for it description, and allows the content to send “statements” based around [actor] [verb] [object], or “I – did – this” to a Learning Record Store (LRS) which can be part of Learning Management System but can live on their own or as part of another system. This was the first step for **Separation of concerns** in e-learning.

[4]Adobe Captivate

Learning Record Store LRS

A Learning Record Store is an external to course application that receives and sends data in JSON format from and to course runtime - it is an essential component in Experience API process flow. What's a big difference is that the specification does not tell how does course is being played (course runtime), it just defines that runtime does communicate with the interface (LRS) though xAPI Statements. The statements are open to extend, each implementation can introduce their own statements.

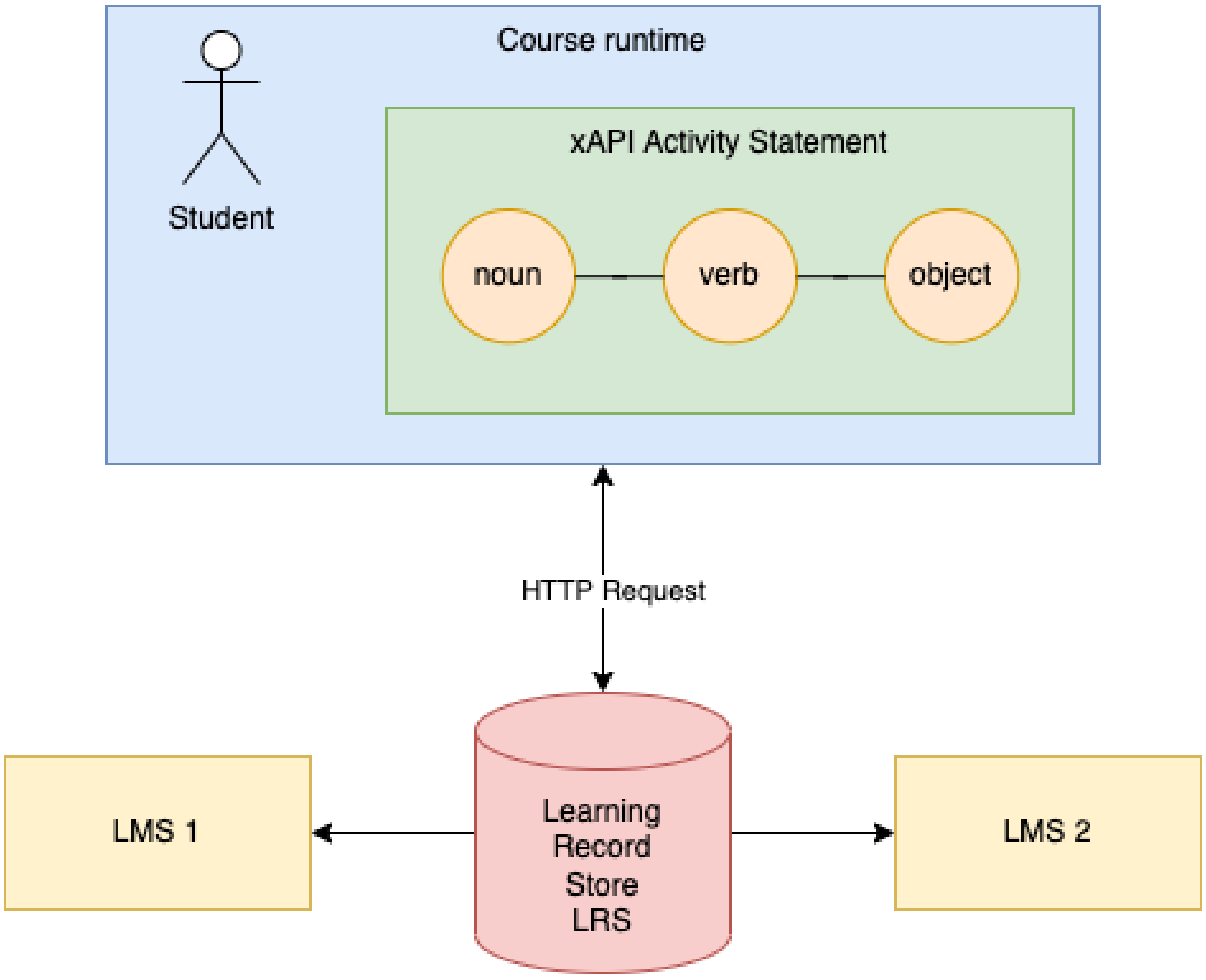

Figure 4: Experience API (xAPI) process flow with Learning Record Store (LRS)

Limitations parts of current standards

- no offine (needs a server)
- same domain
- no speparationn betrween layers
- when designed there was no mobile devices
- Assignable Unit (AU) must have launchURL that basically is course starting point
- why cmi5 is not enough (http://aicc.github.io/CMI-5_Spec_Current/flows/lms-flow.html)
- why xapi is good

Bad parts

Separation of concerns

> A design principle for breaking down an application into modules, layers, and encapsulations, the roles of which are independent of one another. [5]

What is headless

PWA, Serverless, JSON (vs XML)

Ulam Format

Why separaion is good https://en.wikipedia.org/wiki/Model%E2%80%93view%E2%80%93controller

> Universal Learning Asynchronous Model

Good parts from cmi5 & xapi

- xapi verbs
- au
- json > xml

> Tabela z porownanie ulam vs cmi5

HEadless

Players

Import/Export

Offline

The Conclusions

In general a short summarizing paragraph will do, and under no circumstances should the paragraph simply repeat material from the Abstract or Introduction. In some cases it's possible to now make the original claims more concrete, e.g., by referring to quantitative performance results.

Future Work

This material is important – part of the value of a paper is showing how the work sets new research directions. I like bullet lists here. (Actually I like them in general.) A couple of things to keep in mind:

- If you're actively engaged in follow-up work, say so. E.g.: "We are currently extending the algorithm to... blah blah, and preliminary results are encouraging." This statement serves to mark your territory.
- Conversely, be aware that some researchers look to Future Work sections for research topics. My opinion is that there's nothing wrong with that – consider it a compliment.

[5] Blockchain Networks: Token Design and Management Overview NISTIR 8301. National Institute of Standards and Technology

The Acknowledgements

Don't forget them or you'll have people with hurt feelings. Acknowledge anyone who contributed in any way: through discussions, feedback on drafts, implementation, etc. If in doubt about whether to include someone, include them.