EAI Endorsed Transactions
on Creative Technologies
Research Article **EAI.EU**

Towards a novel and LMS-free Pervasive Learning System exploiting the Experience API

Spyros Panagiotakis[1,*], Koralia Papadokostaki[1], Kostas Vassilakis[2] and Athanasios Malamos[1]

[1] Department of Informatics Engineering, Technological Educational Institute of Crete, TEI Campus, Estavromenos, 71004 Heraklion, Crete, Greece

[2] Department of Electrical Engineering, Technological Educational Institute of Crete, TEI Campus, Estavromenos, 71004 Heraklion, Crete, Greece

Abstract

Experience API (xAPI), the evolution of SCORM (Sharable Content Object Reference Model) in e-learning content specifications, stably gains ground in education, defense and training. In this paper we shortly describe the xAPI specification, compare it to its predecessor, the SCORM, and present our novel, Pervasive Learning System, which exploits the features offered by xAPI. Our implementation provides adapted learning content to the user according to his past activities, which are monitored via xAPI. Furthermore, our system provides the course instructor with a Dashboard and several powerful Learning Analytics. The latter offer insights into the learner's use of the course, average scores, actions performed and so on. Our Learning System does not involve the deployment of any Learning Management System. It is a flexible integrated system that may track and monitor the actions of users without the implication of external applications, whereas the statements may origin from diverse, scattered platforms.

Keywords: Experience API, Tin Can API, xAPI, LRS, SCORM, mobile learning, pervasive computing, Adaptive learning, e-learning

Received on 30 May 2018, accepted on 06 July 2018, published on 18 July 2018

doi: 10.4108/eai.13-7-2018.156383

[*] Corresponding author. Email:spanag@ie.teicrete.gr

1 Introduction

As mobile devices gain constantly popularity and conquer our everyday lives and habits, online learning has shifted from traditional LMSs (Learning Management Systems) to an everyday and everywhere process [1] [2] [3]. Games, augmented reality applications, virtual worlds, streaming platforms and social networks may nowadays serve as training or teaching sources, while mobile equipment of different types plays the role of the medium [4]. This new mobile learning model offers surplus value to a user's learning environment [2],[5].

In this context of unsupervised learning, however, the benefits for learners will be greater if a transparent way of monitoring and guiding their learning progress exists. Such a mechanism shall keep a track of the learning activities of individuals, so learning profiles are built and customized educational resources are provided to them according to their past activities. Considering the plethora of available learning resources today, it is obvious that this mechanism should be distributed and provide a generic and standardized way for reporting learning activities. According to this plan, the available educational resources should be able to communicate with various distributed backend repositories of learning actions, using a standardized protocol and vocabulary. The exchange of such information should be also possible between the repositories, so a common knowledge base of individual learning activities is built. This base could be potentially used by various learning providers, so educational content adapted to the learning level of an individual is offered. The latter entails that an intelligent learning agent should exist that will take into account the reported learning activities of an individual in the repositories, to orchestrate its further learning.

Identifying the need to support mobile and non-traditional sources of learning and to host tracking information for each user's learning curve, ADL (Advanced Distributed Learning) [6] has developed a new specification, the Experience API [1], [7], which is also known as xAPI. In a few words, xAPI is a "platform and content agnostic" [7] tool that can dynamically track and store activities from any platform or software system, as those aforementioned.

In this paper we will shortly address the concept of xAPI, explore its basic infrastructure and compare it to its predecessor, SCORM [8]. Moreover, we will present our novel implementation which consists of a) learning content: two standalone courses that seamlessly communicate without a Learning Management System making use of xAPI, b) backend Learning Record Store (LRS), which is responsible for receiving, storing, and providing access to xAPI activity streams, c) intelligent learning orchestration: our agent communicates with the underlying repository of learning activities to adapt the offered content according to former learning activities and d) Teacher's dashboard that provides learning analytics on the students' progress and their difficulty on taking the course. To the best of our knowledge there is no other LMS- independent application that enables adaptive sequencing path according to a user's previous activities and the interaction of two courses through xAPI. Our xAPI enabled application does not need the setup and configuration of an LMS and users only use their email in order to enter a personalized course. This is achieved by employing the JavaScript libraries which implement xAPI to store and track learning activities.

This article extends our work described in [9]. In the latter we had presented an LMS-free Learning System based on xAPI that captures learner's activities and automatically modifies the learning path of a course according to the learner's history. Now we advance one step further introducing our integrated solution for pervasive learning. The last development incorporates a Learning Analytics tool. Hence, it processes the data generated by the xAPI-enabled course and yields powerful learning reports for the course and the learners. The querying form, tables and interactive charts provided are priceless to the instructor and course designer for the overall assessment of the course and learning material and the improvement of the learning process.

The rest of this paper is structured as below: in Section 2 the Experience API (xAPI) specification is introduced and a comparison between xAPI and SCORM is attempted, while in Section 3 xAPI's uses in education are presented. In Section 4 our novel implementation is described and in Section 5 a deployment of our system in real conditions is presented. Finally, in Section 6, conclusions upon our implementation and plans for future work are discussed.

2 xAPI: Inside the Specification

2.1 ADL's SCORM: the predecessor of Experience API

The ADL (Advanced Distributed Learning) Initiative is a US government program aiming to augment flexible, lifelong learning through the use of technology [6]. It is widely known for its SCORM specification [8], introduced in 2001, and revised up to the SCORM 2004 4th Edition. SCORM intended to overcome the major problems of interoperability and reusability of learning content. Before SCORM was proposed, the process of tracking the learner's progress was tailor-made for each platform; if the company or foundation changed its LMS, the tracking process had to be redesigned and re-implemented. With the use of the SCORM model, the learning content is packaged into a format which can be transferred through various Learning Management Systems (LMSs) [5], [8], [10] accomplishing thus not only interoperability, but also reusability, traceability and longer lifecycle.

Although SCORM was welcomed with applauses, adopted, supported and compliant with popular LMSs and perhaps the most "widely used e-learning format" [11], rapid rise of technology caused its glory to gradually fade away. To start with, SCORM is tightly connected to the LMS ("LMS-centric" as stated in [12]) and cannot exist autonomously [2]. However, in a constantly changing world, where learning happens also beyond the LMS and through mobile devices (tablets, smartphones, smart television sets even gaming consoles), there is a need for support of informal and ubiquitous education [2], which is neatly described with the motto "Learning is happening everywhere" [1].

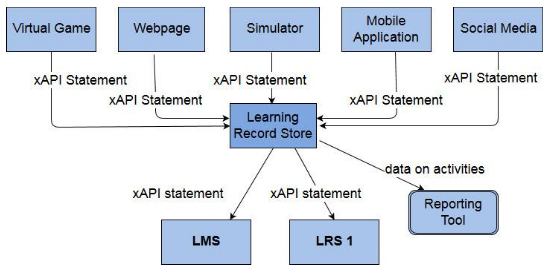
That was the vision Learning-Education-Training Systems Interoperability (LETSI) tried to realize in 2008, when it started investigating the requirements of the next generation of SCORM (SCORM 2.0). After lots of whitepapers and suggestions [13], ADL focused on standardized experience tracking capabilities and in 2010 a Broad Agency Announcement (BAA) project evolved: the "Experience API." Rustici Software - the company that undertook the project - renamed it to "Project Tin Can" as this term implied the two-way conversation between the company and the e-learning industry [1] ,[7] and today the two terms are synonymous.

2.2 Experience API- Understanding the Basics

The new technical specification called Experience API (also known as xAPI or Tin Can API) was launched in 2012, under the version 0.9 and up to today several versions have been launched adding extra functionality and clarifying many issues. The current version at the time of writing is version 1.0.3 and was launched in September, 2016 [14]. The xAPI was and remains an open source, learning technologies interoperability specification that describes tracking of learner activities and experiences between technologies [15]. It is licensed under the Apache License, Version 2.0 and is widely updated and supported by the community [7].

Based on the concept of activity streams that popular social media, such as the Facebook and the Twitter, already use, xAPI can capture learning activities in the form of activity streams originating from various means and contexts [2],[10]. These records are transferred and kept in a server, called Learning Record Store (LRS), which is responsible for receiving, storing, and providing access to them. The xAPI not only specifies the structure of the streams of learning experiences, but also defines the details for their transfer and storage [10],[15]. The core elements of the specification, the (learning) activity streams are called "Statements" (xAPI statements) and describe how the learner interacted with an object, e.g. whether a learner completed a course, accomplished a quiz or watched a video. In their basic format they follow the structure of <*Actor*, *Verb*, *Object*>, but as the object can be of various types this structure can be extended by adding extra optional information, such as the result of a quiz, the timestamp of the activity or the context of an activity [12],[16]. The statements are identified by a unique UUID (Universally Unique Identifier) and are transferred and stored in JSON (JavaScript Object Notation) format.

xAPI supports a predefined Vocabulary, comprising of a large set of Verbs and Activity Types to support various cases. Verbs include *attempted*, *failed*, *experienced*, *shared*, while Activity Types may be *simulation*, *course*, *media*, *meeting*, *assessment* or *file* among others. Both sets are updated regularly and extended [17].



.

Figure 1. xAPI supports a distributed architecture, where statement streams can originate from diverse sources and may be delivered to several endpoints.

An LRS is not only a data store for statements, but it can also allow the retrieval of these statements by external applications and serves as a provider of these statements to be aggregated and analyzed. It may provide the source for data aggregation and analytics and can be the repository for extraction of precious information from basic statements [2],[10]. Apart from reporting and analytics, an LRS can be a valuable tool for personalized approaches, as activities stored in the format <who did what> can be easily processed. Additionally, it can host activities from various sources and that is its main advantage: whether the statement comes from a serious game, a mobile application or a webpage it can be stored under the same format in the LRS. Finally, an LRS can exchange data with other LRSs or LMSs, meeting thus different requirements. Figure 1 illustrates the versatility of xAPI and the vast spectrum of applications it cooperates with.

2.3 xAPI vs. SCORM

xAPI was advertised as the evolution of SCORM and similarities should be self-evident. Nevertheless, xAPI is a much wider technology than SCORM, can be used in various circumstances and has many advantages compared to SCORM. Firstly, in order to use SCORM, the learning contents should be delivered in SCORM packages, which can be a serious limitation for the content developer. In xAPI, though, learning activities or contents can be totally independent of data formats [18], as simple web content can be a learning activity and libraries or applications implementing the xAPI specification can provide the infrastructure for the delivery of the statements to an LRS. This makes tracking activities from various sources a reality; statements concerning the same learner may originate from webpages, mobile applications, simulators, virtual games or social networking tools [1],[2]; all these diverse technologies can be used as training systems and data from them should end up in the same storage unit, the same LRS. The xAPI extends learning environments further than SCORM and provides independence from LMSs, fulfilling this way the vision of 'lifelong learning', since learning can happen everywhere. Additionally, as xAPI is based on the delivery of statements relative to the content and not the content itself, they are easier to implement and give the content-developers flexibility concerning the content and the hosting of the content. For example, the content may be offline and xAPI may deliver the statements to the LRS through an occasional connection to the Internet [2],[12]. This is a major advantage for xAPI, as the learner need not be constantly online, but may still contribute to the LRS with the activities that he accomplished in form of statements. Moreover, xAPI may prove to be a priceless mechanism in the hands of data analysts, as it can cooperate well with Business Analytics (BI) and reporting tools, contrary to LMSs, the traditional hosts of SCORM content [1],[19] [18]. Business Intelligence focuses on strategic exploit of the data kept by a business to lead to important decision making [4], [19]. Finally, xAPI may simultaneously integrate with one or several LRSs, and optionally with an LMS offering this way extra value to the administrator of the data.

The main differences between xAPI and SCORM are presented synoptically in Table 1.

Table 1. The main differences between xAPI and SCORM

	SCORM	xAPI
Activity tracking	from e-learning courses	from e-learning courses, mobile applications, simulators, virtual games or social networking tools
Learning contents	should be delivered in SCORM packages	learning activities or contents can be totally independent of data formats
Availability of content	Need be online	may be offline
LMS- dependency	LMS- dependent	LMS- independent
Cooperation with Business Analytics (BI) tools	Does not cooperate well with BI	Cooperates with LRSs, LMSs, BI and reporting tools

3 xAPI in Education

xAPI broadens e-learning and its potentials, by adding tracking to various learning activities in a seamless manner. It is suitable for tracking learning activities that happen in a learning system, i.e. an LMS or an online course, but it is also ideal for recording learning activities that are not hosted in traditional learning systems. As Internet becomes the main repository of knowledge nowadays, online resources are potential sources of informal learning. In real life informal learning can happen everywhere and anytime and informal e-learning should follow these trails.

With Internet and mobile devices, YouTube videos, serious games, simulations or posts on social media can provide valuable knowledge to the learner; with the use of xAPI and its implementations, all these learning activities can be captured and may contribute to the definition of each learner's personal profile. Till now, only knowledge that was delivered through formal e-education could be recorded, now tracking informal learning may give us additional information upon the learner, the content and the learning process. Keeping a record of an individual's learning experiences can play an important role in providing him with the proper content in the most efficient manner, which is the goal of Adaptive Learning/ training [20]. Building an adaptive learning system may alter the content to meet the learner's needs or might change the way the content is presented according to the learner's profile [10],[20],[21].

From the perspective of Learning Analytics, where educational data is collected and analyzed aiming in the discovery of patterns in learning process or problems in student performance, xAPI is indeed a very promising technology [19]. In the five stages of collecting, reporting, predicting, acting and refining [19], xAPI can pioneer in collecting data from various sources (not necessarily LMSs) and provide aggregate or summarized data to third-party tools for reporting and predicting [19]. Extracting Analytics and therefore knowledge from gathered data can be used by students as self-awareness tools; by teachers for self-evaluation and detection of issues in their classroom or as a motive for improvement, while schools may use tools for their planning, decision-making and as part of Business Intelligence [22],[23].

Furthermore, xAPI can promote collaborative learning through the use of collaborative applications, social media or even serious or virtual games. Using it may augment teamwork and may convert e-learning from personal learning to team-learning [2],[24].

4 Implementing an Adaptive Learning System with xAPI

4.1Related work

Through an extensive research in bibliography, we have not found applications that make use of xAPI in order to develop interactive applications without the use of an LMS. Some applications make use of xAPI statements gathered in an LRS for monitoring and extracting data. The Oregon Trail Game, for example, is a classic e-learning game that has been updated to track learner activities through xAPI [5], however does not support any interactivity based upon the statements. Other applications require LMS integration. The LIME project includes a recommender system based upon previous activities of the user [25], but it is not made clear, whether the recommendation is interactive based upon the specific user's activities. Moreover, it seems completely tight to the LMS [5] and rules in LIME cannot operate upon individual LRS records, but only upon averages and aggregated data, 'which offer a more equalized view of the learner situation' [25], [26]. The Mobler Cards App [16] introduces an application with flashing cards that incorporates an LRS and bases its decisions upon previous activities of the user. Although it requests the LRS in order to orchestrate the sequence of the question, the LRS only stores activities based on the application itself. AubiLearn [27] is a research project aimed at adapting multimedia content to different contexts- it provides adaptation according to the device the user uses in order to implement u-learning, however it does not provide learning analytics not does it provide learning material from different sources . Similarly, an adaptive u-learning system has been provided in [28], which combines adaptation with User's Experience but employs an LMS [28]. Similar researches have been implemented to manifest the importance of adaptive u-learning in specific lessons or contexts [29], [30] .

However, in our research we have not come across an LMS- independent application that enables adaptive sequencing path according to a user's previous activities and the interaction of two courses through xAPI and provide LA at the same time. Towards this direction, we have implemented an xAPI enabled application that needs not the setup and configuration of an LMS and where users only use their email in order to enter a personalized course. The present paper extends the work presented in [9] by introducing learning analytics in our work and in specific a dashboard tool for facilitating assessment and reporting by the teacher. Furthermore, thorough presentation of our deployment is attempted.

4.2 Architecture of the Implementation with the Use of xAPI

In this section we will describe our Ecosystem for adaptive learning and its infrastructure. The architecture is illustrated in Figure 2 and consists of three main modules:

- The learning *content* which enables tracking activities with xAPI
- The *LRS* which stores the xAPI statements
- The *Dashboard* which provides aggregate and visualized information to the teacher

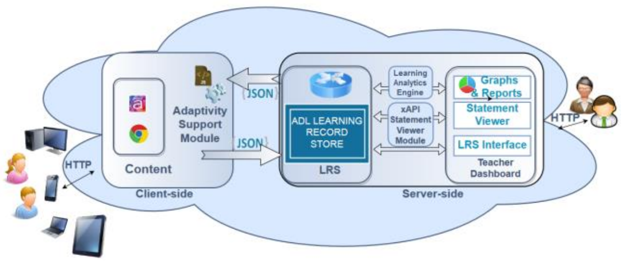


Figure 2. The Architecture of our System

The user may use a computer or a mobile device in order to interact with the content. The content tracks specific activities of the user and sends them to the LRS. An adaptivity support module is added in the client-side module in order to adapt learning content according to the activities recorded in the LRS, regarding the user. The instructor on the other hand, watches the activities of the user with visualizing and monitoring tools provided from our Dashboard. For the reports to be produced and the statement viewer to provide access to the statements a Learning Analytics Engine has been developed and the xAPI statement viewer module has been adapted to our needs.

4.3 Architecture of the Learning Content with the Use of xAPI

Our vision was to take advantage of the capabilities of xAPI in order to create an adaptive learning system [10],[20] which will adjust its content according to the previous activities a learner has accomplished. For the learning system to be effective and accurate we had to use the online delivery of the statements from the activity provider towards the LRS. Additionally, our learning system should have access to the statements in the LRS, in order to change the content accordingly. In our case, the Activity Provider and the Activity Consumer are parts of the same application. The intermediate service, the LRS, stores the statements and acts as a server to our client- server application. However, the decision-making is made in the client as the course runs on the browser.

Our implementation is addressed towards fifth and sixth grade students of Primary Education. It is a brief course on spreadsheets that includes a short introduction on their use and usability and demonstrates basic concepts about sheets, cells and their format. As spreadsheets belong to the same office suite with word processors and presentations software, students may be familiar with some features of this software. Therefore, our learning content consists of two independent courses which are two separate webpages. Course 1 is about text formatting and may be included in word processors, spreadsheets and presentations, whereas Course 2 provides learning content for spreadsheets.

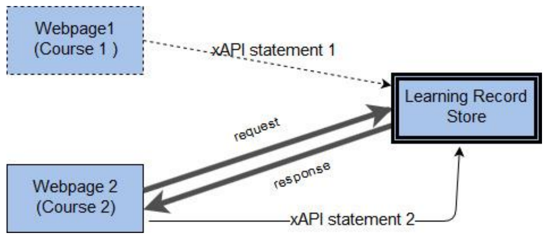


Figure 3. The architecture of our learning content. A dashed box suggests that the course is not required and dashed lines that the relevant statement may not be sent to the LRS. Thick arrows show the communication between the course and the LRS regarding the existence of xAPI statement.

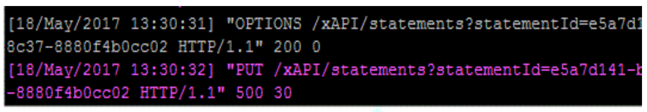
When the learner accomplishes Course 1, a statement is sent to the LRS. When he launches Course 2, the course 'asks' the LRS if that statement exists in the LRS (Figure 3) and if it does, it does not show the content which was included in Course 1. If the statement does not exist - i.e. the learner has not come across text formatting - the content regarding the Course 1 is displayed to the learner. For instance, if Course 1 was offered as part of a word processing lesson but the student was absent at that time or failed that course, he should revise this content and therefore our implementation in Course 2 should provide him with the information included in the Course 1. This way, our implementation offers personalized pathway to the learner according to his previous activities and adapts the content according to the learner's history and needs.

One of the originalities of our implementation is that it does not need account passwords or credential management and it is based on simple virtual emails. This serves the needs of our implementation since our work is aimed to very young learners. Of course, xapi credential management options could have been used to fulfill more advanced requirements.

4.4 Implementation of the Learning Content

For our implementation, an instance of ADL's Open Source Learning Record Store (LRS) [31] was installed in an UBUNTU server. Inventors of xAPI along with the community have developed libraries in several languages, e.g. JavaScript, C, Java, PHP and Python in order to implement the xAPI specification. The library in JavaScript, called Tin Can.js, is constantly supported and updated by developers and seemed the ideal solution for us to implement our course.

The courses are webpages (.html files) with JavaScript code which performs the communication between the courses and the LRS. In order to develop user-friendly courses that would be enriched with attractive interface, multimedia content and interactive quizzes, we used a demo version of Articulate Storyline 2 [32], which is a popular software for creating learning content. Articulate Storyline 2 efficiently supports Tin Can API, but its integrated features provide just one-way delivery of statements, i.e. from the course to the LRS and not vice versa. Hence, this was not enough for our implementation. Therefore, we decided to use the Tin Can API JavaScript library [33], so we can provide the necessary bidirectional communication between html courses and LRS. Our courses in Articulate Storyline 2 were augmented with JavaScript code calling functions to make delivery of the statements to and from the LRS possible. As stated earlier, Course 1 is an optional part which might have been attempted in the past by a student and may belong to a different class. If it is completed successfully, our implementation sends a statement (via the JavaScript Tin Can function sendStatement) to the LRS with the indication that the student has completed Course 1. The Tin Can JavaScript function sendStatement is implemented via Restful HTTP PUT (or POST) method [15]. Figure 4 shows the statement that is sent via a PUT function towards the LRS.



```
[18/May/2017 13:30:31] "OPTIONS /xAPI/statements?statementId=e5a7d1
8c37-8880f4b0cc02 HTTP/1.1" 200 0
[18/May/2017 13:30:32] "PUT /xAPI/statements?statementId=e5a7d141-b
-8880f4b0cc02 HTTP/1.1" 500 30
```

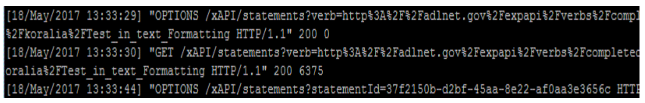
Figure 4. xAPI supports a distributed architecture, where statement streams can originate from diverse sources and may be delivered to several endpoints.

The statement is stored in our instance of ADL's LRS and is shown in the JSON format that follows.

```
{ “verb": {
 "id":
"http://adlnet.gov/expapi/verbs/completed",
     "display": {
        "und": "completed"     }    },
   "version": "1.0.2",
   "timestamp": "2017-05-18T17:30:32.680Z",
   "object": {
      "id": "http://koralia/Test_in_text_Formatting",
      "objectType": "Activity"},
   "actor": {
      "mbox": "mailto:koraliap@test.com",
      "objectType": "Agent"},
   "stored": "2017-05-18T17:30:31.282349+00:00",
   "result": {
      "completion": true,
"score": {
         "scaled": 1 },
      "success": true    },
   "id": "e5a7d141-b4d2-4acd-8c37-8880f4b0cc02",
   "authority": {
      "mbox": "mailto:mtp130@edu.teicrete.gr",
      "name": "koralia",
      "objectType": "Agent"   }}
```

The short format of the statement is:
mailto:koraliap@test.com (*Actor*)
http://adlnet.gov/expapi/verbs/completed (*Verb*)
http://koralia/Test_in_text_Formatting (*Object*)
and follows the structure of <*Actor, Verb, Object*> which was mentioned earlier in this paper.

When the student attempts Course 2, which might be in a posterior point in time, the student need not repeat Course 1. Therefore, our implementation sends – via JavaScript Tin Can *getStatements* function– a request towards the LRS asking whether the statement with the indication that the student has completed Course 1 exists. Again Tin Can JavaScript function *getStatements* is implemented via a Restful HTTP *GET* method (illustrated in Figure 5) [15].



```
[18/May/2017 13:33:29] "OPTIONS /xAPI/statements?verb=http%3A%2F%2Fadlnet.gov%2Fexpapi%2Fverbs%2Fcompl
%2Fkoralia%2FTest_in_text_Formatting HTTP/1.1" 200 0
[18/May/2017 13:33:30] "GET /xAPI/statements?verb=http%3A%2F%2Fadlnet.gov%2Fexpapi%2Fverbs%2Fcompleted
oralia%2FTest_in_text_Formatting HTTP/1.1" 200 6375
[18/May/2017 13:33:44] "OPTIONS /xAPI/statements?statementId=37f2150b-d2bf-45aa-8e22-af0aa3e3656c HTTP
```

Figure 5. An example of the request of a statement towards the LRS.

The LRS responds and if the requested statement is found, our implementation skips this part and continues with new content (Figure 6). If the statement is not found, our implementation shows the content of the first course as well as the new content (Figure 7). To avoid bottlenecks and delays, the request to the LRS has been sent at the beginning of the second course and the response is stored in a local variable.

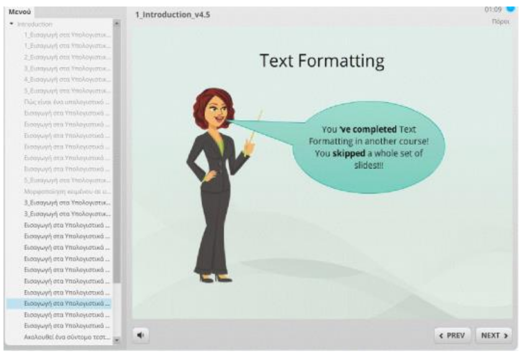

1_Introduction_v4.5
Text Formatting
You 've completed Text Formatting in another course! You skipped a whole set of slides!!!
PREV
NEXT

Figure 6. The user has previously completed Course 1 and automatically skips the slides concerning this content. In the menu on the left, the slides concerning the content of Course 1 appear to have been skipped.

The exported courses are in html format and can be hosted in any website together or independently. They are standalone applications that need not an LMS to integrate, but instead need an LRS to communicate with. Along with the course files, the tincan.js file should be uploaded in the web servers, whereas the browser should support JavaScript.

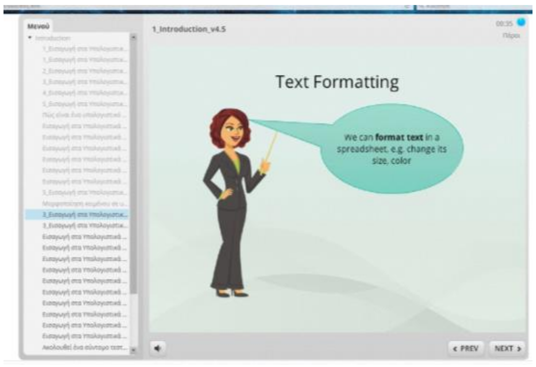
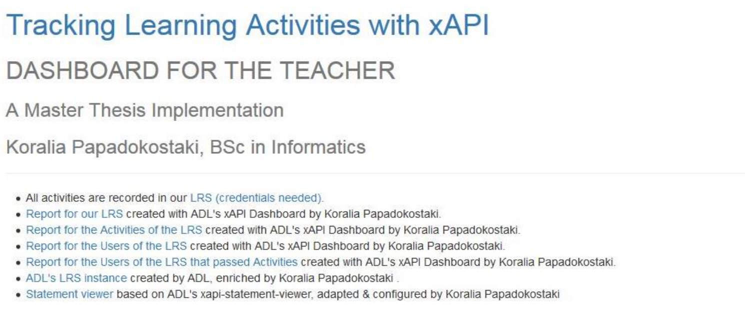

1_Introduction_v4.5
Text Formatting
We can format text in a spreadsheet, e.g. change its size, color
PREV
NEXT

Figure 7. If the user has not completed Course 1, he watches the slides concerning relevant content.

4.5 Teacher's Dashboard

Our implementation is equipped with a powerful monitoring tool, named "Teacher's Dashboard". The Dashboard provides the teacher with three powerful tools for monitoring and visualization of the user's activities. As earlier mentioned, these tools may be used in order to provide learning analytics. More specifically they can be used to measure user's activity or performance, to track the impact of the content upon the learner, or even to investigate the difficulty of several parts of our content. As the content may come from various sources the frequency of each medium may as well be measured. To this direction, our implementation's dashboard (Figure 8) provides:

- Link to the built in ADL LRS interface
- A statement viewer
- Visualization tools (charts, pies and reports)



Tracking Learning Activities with xAPI

DASHBOARD FOR THE TEACHER

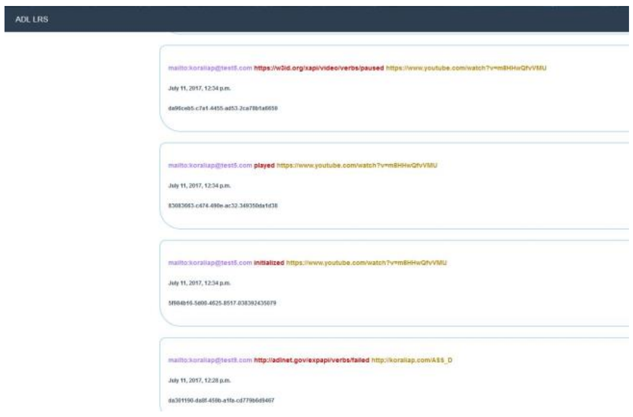
A Master Thesis Implementation

Koralia Papadokostaki, BSc in Informatics

- All activities are recorded in our LRS (credentials needed).
- Report for our LRS created with ADL's xAPI Dashboard by Koralia Papadokostaki.
- Report for the Activities of the LRS created with ADL's xAPI Dashboard by Koralia Papadokostaki.
- Report for the Users of the LRS created with ADL's xAPI Dashboard by Koralia Papadokostaki.
- Report for the Users of the LRS that passed Activities created with ADL's xAPI Dashboard by Koralia Papadokostaki.
- ADL's LRS instance created by ADL, enriched by Koralia Papadokostaki
- Statement viewer based on ADL's xapi-statement-viewer, adapted & configured by Koralia Papadokostaki

Figure 8. The Teacher's Dashboard

ADL LRS provides a statement viewer that can be used for the monitoring of the statements. It shows a list of statements with chronological order from the most recent to older ones (Figure 9). Unfortunately ADL LRS's interface does not support dynamic searching or filtering on the statements.



ADL LRS

mailto:koraliap@test5.com https://w3id.org/xapi/video/verbs/paused https://www.youtube.com/watch?v=m8HHwQfvVMU
July 11, 2017, 12:34 p.m.
da96ceb5-c7a1-4455-ad53-2ca78b1a6658

mailto:koraliap@test5.com played https://www.youtube.com/watch?v=m8HHwQfvVMU
July 11, 2017, 12:34 p.m.
83083663-c474-490e-ac32-349350da1d38

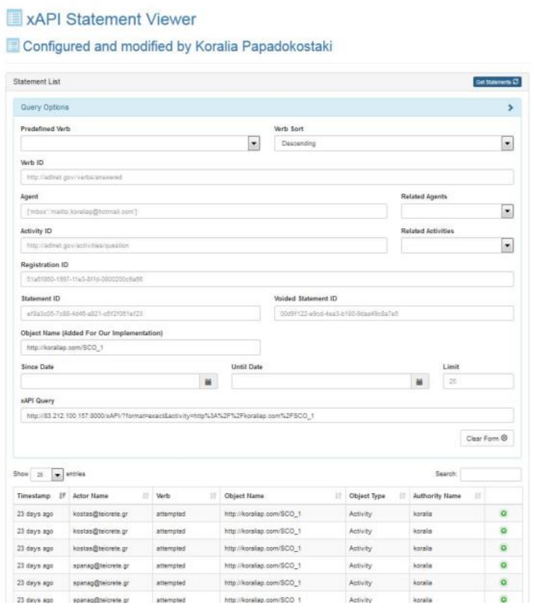
mailto:koraliap@test5.com initialized https://www.youtube.com/watch?v=m8HHwQfvVMU
July 11, 2017, 12:34 p.m.
5f984b16-5e00-4625-8517-038392435079

mailto:koraliap@test9.com http://adlnet.gov/expapi/verbs/failed http://koraliap.com/ASS_D
July 11, 2017, 12:28 p.m.
da301190-da8f-458b-a1fa-cd779b6d9467

Figure 9. The ADL LRS interface for the administrator

Due to this fact, we had to complement our implementation with a multifunctional statement viewer. For that reason, we used the ADL's xAPI-Statement-Viewer [34], which provides fields for searching and makes the display of xAPI Statements more practical. In more specific, the ADL's xAPI-Statement-Viewer pulls xAPI Statements from a LRS using the xAPI wrapper [35] and displays them in a user-friendly table, whereas it also offers filtering and sorting options. The filters provided concern the verb, the mail of the user, filters for the date of the statement etc.

However, in order to provide more options in filtering, we modified the original ADL's xAPI-Statement-Viewer and added an extra filtering field which refers to the activity name. This way we may easily extract the conclusion as to who interacted with a specific object (Quiz, video or interaction). The xAPI statement viewer was also configured with the credentials for our LRS and with verbs where the display field was not "en-us" (Figure 10).


xAPI Statement Viewer

Configured and modified by Koralia Papadokostaki

Statement List — Get Statements

Query Options

Predefined Verb:
Verb Sort: Descending
Verb ID: http://adlnet.gov/verbs/answered
Agent: {"mbox":"mailto:koraliap@hotmail.com"}
Related Agents:
Activity ID: http://adlnet.gov/activities/question
Related Activities:
Registration ID: 51a6f860-1997-11e3-8ffd-0800200c9a66
Statement ID: ef8a3c05-7c88-4d46-a821-c6f21061ef23
Voided Statement ID: 00d9f122-e9cd-4ea3-b180-9daa49c6a7e5
Object Name (Added For Our Implementation): http://koraliap.com/SCO_1
Since Date:
Until Date:
Limit: 25
xAPI Query: http://83.212.100.157:8000/xAPI/?format=exact&activity=http%3A%2F%2Fkoraliap.com%2FSCO_1

Clear Form

Show 25 entries — Search:

Timestamp	Actor Name	Verb	Object Name	Object Type	Authority Name	
23 days ago	kostas@teicrete.gr	attempted	http://koraliap.com/SCO_1	Activity	koralia	⊕
23 days ago	kostas@teicrete.gr	attempted	http://koraliap.com/SCO_1	Activity	koralia	⊕
23 days ago	kostas@teicrete.gr	attempted	http://koraliap.com/SCO_1	Activity	koralia	⊕
23 days ago	spanag@teicrete.gr	attempted	http://koraliap.com/SCO_1	Activity	koralia	⊕
23 days ago	spanag@teicrete.gr	attempted	http://koraliap.com/SCO_1	Activity	koralia	⊕
23 days ago	spanag@teicrete.gr	attempted	http://koraliap.com/SCO_1	Activity	koralia	⊕

Figure 10. The xAPI Statement viewer was modified by us to provide more filtering options

In order to enhance our dashboard with attractive and interactive real-time graphs, we used the xAPI Dashboard [36]. It is an open-source library that provides a quick and easy way to generate graphs from xAPI data, and a powerful query language to manipulate it. It is implemented in JavaScript and makes use of the nv.d3.css library in order to produce a variety of charts. It simplifies the process of extracting meaningful aggregate data from xAPI statements and can generate numerous types of charts and visualizations based on that aggregated xAPI data. It allows developers to run SQL-like queries, filters, and aggregations over xAPI data and can generate numerous types of charts and visualizations based on that aggregated xAPI data.

With the use of xAPI Dashboard, we have created four reports that visualize important information about our content, activities and users.

(i) LRS-related Reports

A report that shows statistics regarding the usage of LRS. It consists of two graphs:

- The first graph illustrates the activity per user and indicates how active each user has been in our Application (Figure 11), by illustrating how many statements have been sent to the LRS, regarding the specific user.

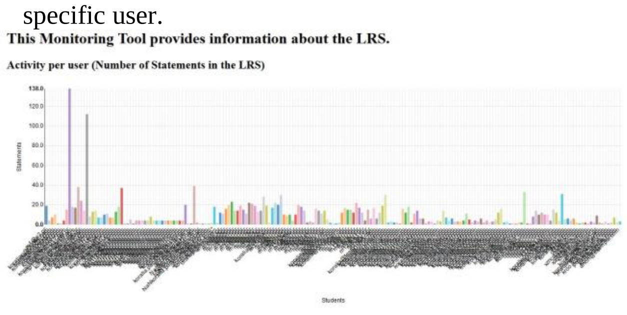


Figure 11. Activity per user in the LRS

- The second graph shows the distribution of score for all activities in our LRS. It illustrates the average score in a pie chart and the percentage of statements with this score. This helps us gather information about the general performance of our students in the LRS (Figure 12).

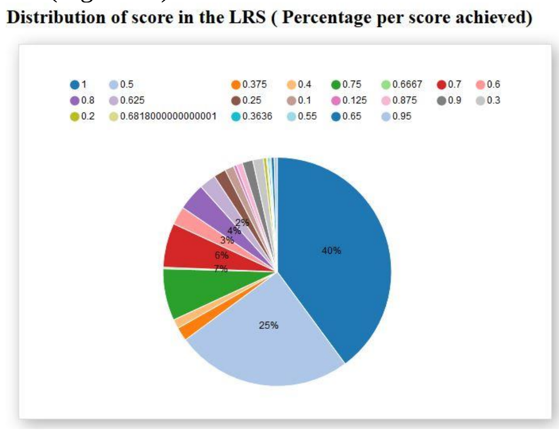
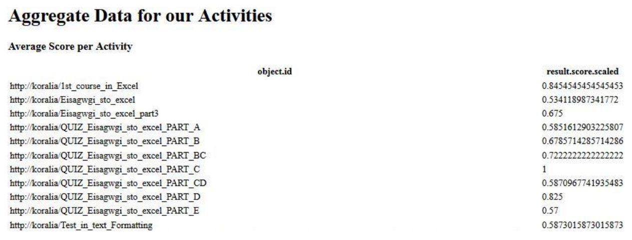


Figure 12. Average score in the LRS

(ii) Activities-related reports

A report regarding specific Activities displays visually information about certain parts of our course, e.g. a quiz. It includes a table and a bar chart with the average score per Activity. This is very useful, as displays the average score of each activity, as the teacher may detect difficult quizzes, vs. easy ones just with one look (Figure 13, Figure 14).


Aggregate Data for our Activities

Average Score per Activity

object.id	result.score.scaled
http://koralia/1st_course_in_Excel	0.8454545454545453
http://koralia/Eisagwgi_sto_excel	0.534118987341772
http://koralia/Eisagwgi_sto_excel_part3	0.675
http://koralia/QUIZ_Eisagwgi_sto_excel_PART_A	0.5851612903225807
http://koralia/QUIZ_Eisagwgi_sto_excel_PART_B	0.6785714285714286
http://koralia/QUIZ_Eisagwgi_sto_excel_PART_BC	0.7222222222222222
http://koralia/QUIZ_Eisagwgi_sto_excel_PART_C	1
http://koralia/QUIZ_Eisagwgi_sto_excel_PART_CD	0.5870967741935483
http://koralia/QUIZ_Eisagwgi_sto_excel_PART_D	0.825
http://koralia/QUIZ_Eisagwgi_sto_excel_PART_E	0.57
http://koralia/Test_in_text_Formatting	0.5873015873015873

Figure 13. Average score per Activity

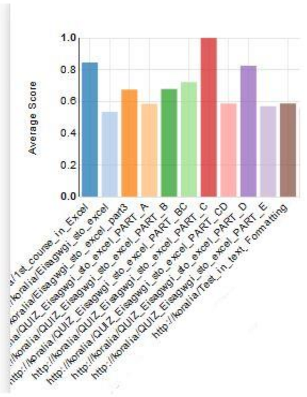
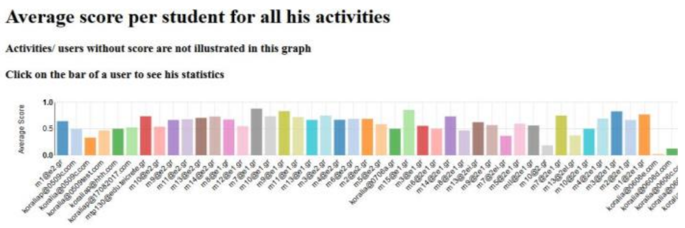


Figure 14. Bar chart with the average score per Activity

(iii) Users-related reports

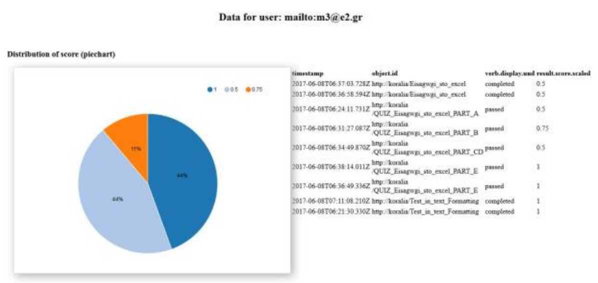
These include an interactive chart (Figure 15) displaying the average score for all users of the LRS. When the teacher clicks on a user's bar, the analytics for the specific user are presented (Figure 16).


Average score per student for all his activities

Activities/ users without score are not illustrated in this graph

Click on the bar of a user to see his statistics

Figure 15. Average Score of students

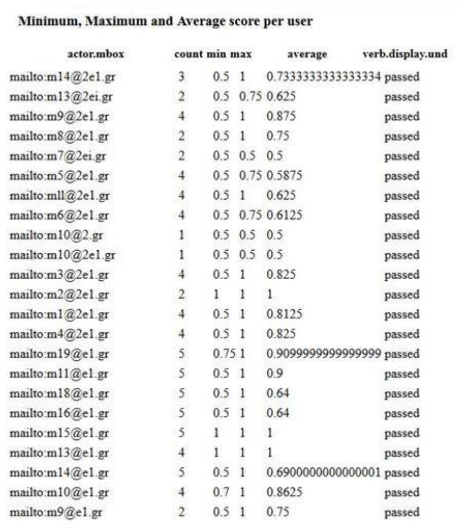

Data for user: mailto:m3@e2.gr

Distribution of score (piechart)

timestamp	object.id	verb.display.und	result.score.scaled
2017-06-08T06:37:03.728Z	http://koralia/Eisagwgi_sto_excel	completed	0.5
2017-06-08T06:36:58.594Z	http://koralia/Eisagwgi_sto_excel	completed	0.5
2017-06-08T06:24:11.731Z	http://koralia /QUIZ_Eisagwgi_sto_excel_PART_A	passed	0.5
2017-06-08T06:31:27.087Z	http://koralia /QUIZ_Eisagwgi_sto_excel_PART_B	passed	0.75
2017-06-08T06:34:49.870Z	http://koralia /QUIZ_Eisagwgi_sto_excel_PART_CD	passed	0.5
2017-06-08T06:38:14.011Z	http://koralia /QUIZ_Eisagwgi_sto_excel_PART_E	passed	1
2017-06-08T06:36:49.336Z	http://koralia /QUIZ_Eisagwgi_sto_excel_PART_E	passed	1
2017-06-08T07:11:08.210Z	http://koralia/Test_in_text_Formatting	completed	1
2017-06-08T06:21:30.330Z	http://koralia/Test_in_text_Formatting	completed	1

Figure 16. Analytics for an individual

Also, a report with aggregated data (average, minimum and maximum scores) about the users' scores (Figure 17) is available.

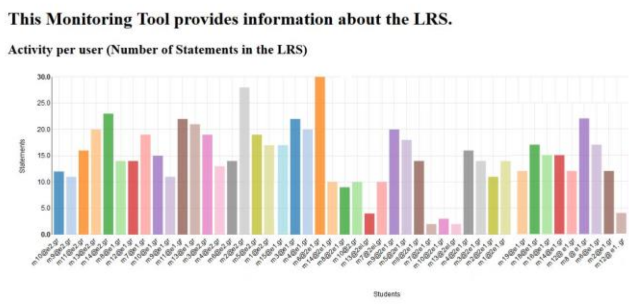

Minimum, Maximum and Average score per user

actor.mbox	count	min	max	average	verb.display.und
mailto:m14@2e1.gr	3	0.5	1	0.7333333333333334	passed
mailto:m13@2ei.gr	2	0.5	0.75	0.625	passed
mailto:m9@2e1.gr	4	0.5	1	0.875	passed
mailto:m8@2e1.gr	2	0.5	1	0.75	passed
mailto:m7@2ei.gr	2	0.5	0.5	0.5	passed
mailto:m5@2e1.gr	4	0.5	0.75	0.5875	passed
mailto:mll@2e1.gr	4	0.5	1	0.625	passed
mailto:m6@2e1.gr	4	0.5	0.75	0.6125	passed
mailto:m10@2.gr	1	0.5	0.5	0.5	passed
mailto:m10@2e1.gr	1	0.5	0.5	0.5	passed
mailto:m3@2e1.gr	4	0.5	1	0.825	passed
mailto:m2@2e1.gr	2	1	1	1	passed
mailto:m1@2e1.gr	4	0.5	1	0.8125	passed
mailto:m4@2e1.gr	4	0.5	1	0.825	passed
mailto:m19@e1.gr	5	0.75	1	0.9099999999999999	passed
mailto:m11@e1.gr	5	0.5	1	0.9	passed
mailto:m18@e1.gr	5	0.5	1	0.64	passed
mailto:m16@e1.gr	5	0.5	1	0.64	passed
mailto:m15@e1.gr	5	1	1	1	passed
mailto:m13@e1.gr	4	1	1	1	passed
mailto:m14@e1.gr	5	0.5	1	0.6900000000000001	passed
mailto:m10@e1.gr	4	0.7	1	0.8625	passed
mailto:m9@e1.gr	2	0.5	1	0.75	passed

Figure 17. Aggregate Data for students

5 Experiences and screenshots from an actual deployment

The Pervasive Learning System of our implementation was used by 46 students of 5th Grade of Primary School in Crete, Greece. They were given an email address (which was virtual) for their actions to be recorded. One group of 25 had previously taken Course 1 and took Course 2, whereas the other one had no previous interaction with Course 1. This was done, in order to test the stability of our implementation and its ability to correctly respond to both scenarios.

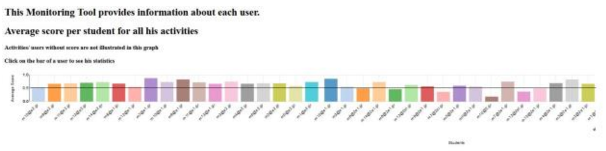

This Monitoring Tool provides information about the LRS.

Activity per user (Number of Statements in the LRS)

Figure 18. The activity for every user of our evaluation

As expected, the users that had successfully completed Course 1 (related to text formatting) in the past, skipped the part of Course 2 that is related to text formatting. The students who had failed in this part had to re-watch the slides regarding this content, similar to the students who had not taken it earlier. In the following we present some charts and reports regarding the reporting capabilities offered by our system, along with learning analytics concerning the data provided.

Figure 18 illustrates the engagement of the users with our course, so we can easily deduct who were very active with the course (e.g. users m6@2e1.gr, m1@e2.gr m14@e2.gr) and who did not interact much with the implementation (e.g. m7@2e1.gr, m10@2e1.gr, m13@2e1.gr). This is partly due to the fact that a group of students took some quizzes more than once in order to improve their scores.


This Monitoring Tool provides information about each user.

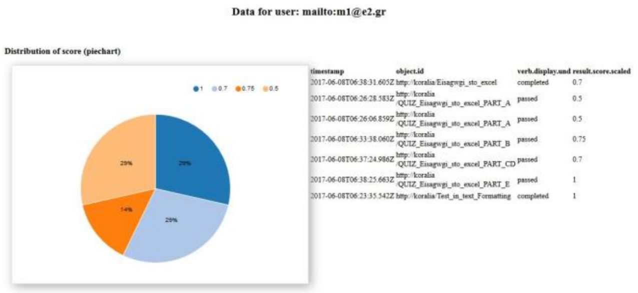
Average score per student for all his activities

Activities' users without score are not illustrated in this graph

Click on the bar of a user to see his statistics

Figure 19. The average score for every user of our evaluation

Figure 19 illustrates the average score of each student, where the black horizontal line indicates the average score for the two courses. With this figure, the tutor may easily recognize the students that fell off the average score of the class (0.58 in our case). In this sample of 35 students, the users who were above the average score were 23 (65,7%), the students very close to the average were 6 (17,15%) and the students lower than the average were 6 (17,15%).


Data for user: mailto:m1@e2.gr

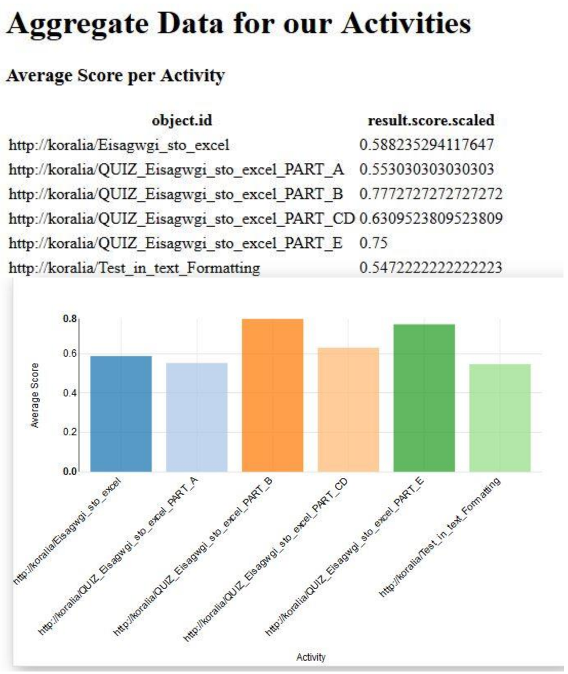
Distribution of score (piechart)

timestamp	object.id	verb.display.und	result.score.scaled
2017-06-08T06:38:31.605Z	http://koralia/Eisagwgi_sto_excel	completed	0.7
2017-06-08T06:26:28.583Z	http://koralia /QUIZ_Eisagwgi_sto_excel_PART_A	passed	0.5
2017-06-08T06:26:06.859Z	http://koralia /QUIZ_Eisagwgi_sto_excel_PART_A	passed	0.5
2017-06-08T06:33:38.060Z	http://koralia /QUIZ_Eisagwgi_sto_excel_PART_B	passed	0.75
2017-06-08T06:37:24.986Z	http://koralia /QUIZ_Eisagwgi_sto_excel_PART_CD	passed	0.7
2017-06-08T06:38:25.663Z	http://koralia /QUIZ_Eisagwgi_sto_excel_PART_E	passed	1
2017-06-08T06:23:35.542Z	http://koralia/Test_in_text_Formatting	completed	1

Figure 20. The score for each statement of the user (pie chart and table)

When the teacher clicks in Figure 19 on a user's barchart he may see analytics regarding the specific user (Figure 20). For example, in Figure 20 we see that user m1@e2.gr was one of the students who at first completed Course 1: "Test_in_text_formatting" and then took the various parts of Course 2: "Introduction to Excel – Eisagwgi_sto_excel". In more specific, from his record we see that he completed Course 1 with score 1 (100% success). Then he proceeded to Course 2, where he attempted PART_A twice (with score 50% at each attempt). Then he completed PART_B with 75% correct answers and PART_CD with 70% correct answers. Finally, he successfully completed the last part of Course 2 (PART_E) with 100% success. The average score with which he completed Course 2 is 70%.

Finally, the average score for every activity of the course can be presented in table and chart format (Figure 21). From the latter chart (Figure 20) we can extract information concerning the difficulty of the parts. Obviously, PART_B (from Course 2) is the easiest (highest average score: 0.77) and Test_in_text_Formatting (from Course 1) is the hardest (lowest average score: 0.54).


Aggregate Data for our Activities

Average Score per Activity

object.id	result.score.scaled
http://koralia/Eisagwgi_sto_excel	0.588235294117647
http://koralia/QUIZ_Eisagwgi_sto_excel_PART_A	0.553030303030303
http://koralia/QUIZ_Eisagwgi_sto_excel_PART_B	0.7772727272727272
http://koralia/QUIZ_Eisagwgi_sto_excel_PART_CD	0.6309523809523809
http://koralia/QUIZ_Eisagwgi_sto_excel_PART_E	0.75
http://koralia/Test_in_text_Formatting	0.5472222222222223

Average Score
0.0
0.2
0.4
0.6
0.8
http://koralia/Eisagwgi_sto_excel
http://koralia/QUIZ_Eisagwgi_sto_excel_PART_A
http://koralia/QUIZ_Eisagwgi_sto_excel_PART_B
http://koralia/QUIZ_Eisagwgi_sto_excel_PART_CD
http://koralia/QUIZ_Eisagwgi_sto_excel_PART_E
http://koralia/Test_in_text_Formatting
Activity

Figure 21. The average score for every activity of the course (table and barchart)

Additionally, when a student's scores (Figure 20) are compared to the courses average score (Figure 21), one can conclude if the user is above, near or below the class average in certain parts of the course. In our example, user m1@e1.gr is way above the class average for Course 2, since he has scored 70% when the class average is just 63%. Furthermore, this user is above the class average in all parts of Course 2 except PART_B, where he achieved 75% with a course average of 77%. This can lead to specific conclusions as to the parts that were hard to understand for this learner. Making use of this information the instructor may focus on this part to help the learner clear the misinterpretations he might have.

Our implementation does not only capture data about the scoring or results of the tests. It also tracks user interaction with the interface of our implementation. For example, in Figure 22 the learner is asked to click the image to see the color palette; when he does, a trigger sends an xAPI statement to our LRS, powering thus interesting conclusions about the interactivity of our implementation and the attention of the learners.

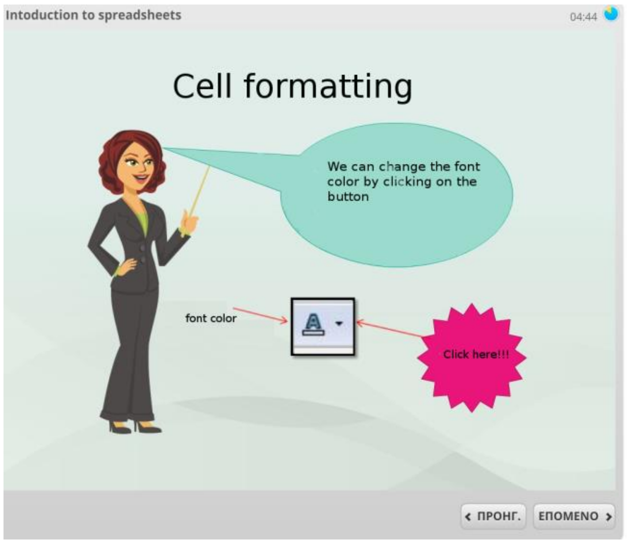
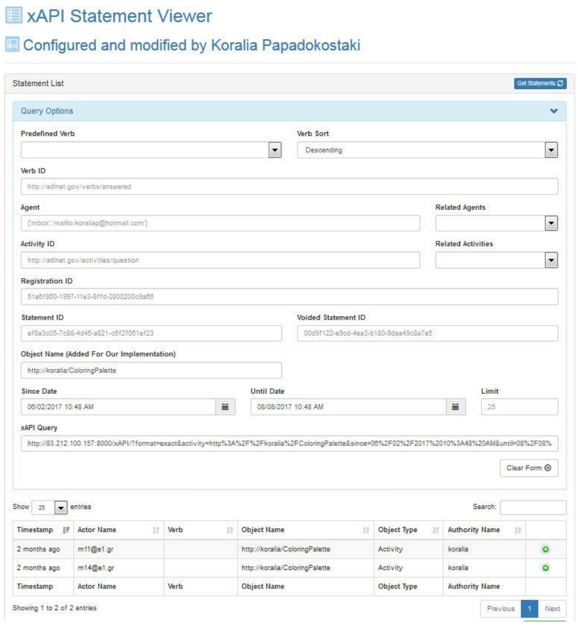

Intoduction to spreadsheets
04:44
Cell formatting
We can change the font color by clicking on the button
font color
Click here!!!
ПРОНГ.
ΕΠΟΜΕΝΟ

Figure 22. A screenshot from the implementation asking learners to click on a button

In Figure 23, the results from the querying form show us that only 2 out of 46 students clicked on the image.


xAPI Statement Viewer

Configured and modified by Koralia Papadokostaki

Statement List

Get Statements

Query Options

Predefined Verb	
Verb Sort	Descending
Verb ID	http://adlnet.gov/verbs/answered
Agent	{'mbox':'mailto:koraliap@hotmail.com'}
Related Agents	
Activity ID	http://adlnet.gov/activities/question
Related Activities	
Registration ID	51a6f860-1997-11e3-8ffd-0800200c9a66
Statement ID	ef8a3c05-7c88-4d46-a821-c6f2f061ef23
Voided Statement ID	00d9f122-e9cd-4ea3-b180-9daa49c8a7e5
Object Name (Added For Our Implementation)	http://koralia/ColoringPalette
Since Date	06/02/2017 10:48 AM
Until Date	08/08/2017 10:48 AM
Limit	25
xAPI Query	http://83.212.100.157:8000/xAPI/?format=exact&activity=http%3A%2F%2Fkoralia%2FColoringPalette&since=06%2F02%2F2017%2010%3A48%20AM&until=08%2F08%

Clear Form

Show 25 entries Search:

Timestamp	Actor Name	Verb	Object Name	Object Type	Authority Name	
2 months ago	m11@e1.gr		http://koralia/ColoringPalette	Activity	koralia	⊕
2 months ago	m14@e1.gr		http://koralia/ColoringPalette	Activity	koralia	⊕
Timestamp	Actor Name	Verb	Object Name	Object Type	Authority Name	

Showing 1 to 2 of 2 entries

Previous 1 Next

Figure 23. The querying form showing which students actually clicked on the button

The main reason we divided our evaluation group into two groups is the fact that we wanted to check the stability of our system and its robustness through massive submissions of xAPI statements. The results were more than satisfying. Although the statements were massively sent to the LRS and the communication between the LRS and the course was two-way, our implementation responded with stability and accuracy; the young students noticed no delay or inconsistency during their engagement with the course. Despite the fact that the architecture of our implementation is distributed, i.e. the course and the LRS are hosted in diverse systems, the response time of our system is satisfactory (about 260 ms, where 200 ms is the roundtrip time between the user and the LRS) and does not affect the responsiveness of our implementation.

From a qualitative point of view, the feedback from the students was enthusiastic and they declared they would like to take similar online courses with short quizzes that would adapt to their background and history. This is a motive for us to continue our research work into producing novel learning systems that are intriguing and effective for students.

6 Conclusions

In this paper we have attempted a short description of xAPI, its functionality and architecture. We have compared it to its predecessor SCORM and have manifested its advantages in a constantly changing world where learning becomes ubiquitous and mobile. As activity streams gain popularity and tracking them offers valuable knowledge, xAPI is being constantly extended, updated and widely adopted in the Learning Industry [37]. Following this trend, we have created an adaptive learning system, making use of xAPI, which modifies its content according to the learner's history, without the use of an LMS.

Our implementation consists of two separate short courses illustrating the features and flexibility of xAPI. Although it was developed in a commercial e-learning authoring tool, it demonstrates the xAPI open source specification and implements the seamless communication between web content and LRS with the use of JavaScript libraries. It only requires an email address for the identification of the user and no extra authentication for him. It does not need the setup of an LMS; on the contrary, it substitutes it and is totally platform independent. Additionally, the content may be scattered across various servers and platforms and each part does not need to coexist with the other parts, offering thus boundless potentials to the distribution of learning content. On top of that, we have developed a teacher's dashboard that provides the instructor with querying form, reports and charts illustrating statistics from the use of our course. These powerful tools can help the teacher assessing his course and students and is only a small sample of the Learning Analytics that xAPI is able to generate. Our software leads the way to innumerable e-learning solutions where the content may be totally free of learning platforms and the learning actions may stem from different platforms; yet the learners may be offered with tailored learning content and personalized learning environment [38], [39]. At the same time the instructors will be able to have access to the data provided and evaluate their teaching overall. With the rapid evolution of Web, this technology will prove valuable and indispensable.

It is within our intentions to expand our application with extra functionalities, enrich it with mobile content, video streaming applications and social media integration. This way we will fully exploit the capabilities of xAPI and make a completely personalized and adaptive course without the use of an LMS, but more importantly with the flexibility and novelty of receiving data from dispersed educational resources.

References

[1] Experience API. (n.d.). Retrieved 26 August 2018, from https://xapi.com/

[2] Murray, K., Berking, P., Haag, J., & Hruska, N. (2012). Mobile Learning and ADL's Experience API. *Connections*, ***12***(1): 45-50.

[3] Panagiotakis, S., Kalogiannakis, M., Ripoll, N., and Vassilakis, K. (2010). Towards Synchronous Tele-Education: Solutions, Trends and Experience Gained. In *proceedings of the 3rd E-Learning Excellence Forum and Conference (eLExForum 2010)*. Dubai, UAE.

[4] Papadokostaki, K., Mastorakis, G., Panagiotakis, S., Mavromoustakis, C. X., Dobre, C., and Batalla, J. M. (2017). Handling Big Data in the Era of Internet of Things (IoT). In C. X. Mavromoustakis, G. Mastorakis, and C. Dobre (Eds.), *Advances in Mobile Cloud Computing and Big Data in the 5G Era* (pp. 3–22). Cham: Springer International Publishing. http://doi.org/10.1007/978-3-319-45145-9_1

[5] Lim, C. P., and Churchill, D. (2016). Mobile learning.

[6] ADL. (n.d.). Retrieved 25 May 2018, from http://www.adlnet.gov/

[7] The xAPI Overview. (n.d.). Retrieved 26 August 2018, from http://www.adlnet.gov/research/performance-tracking-analysis/experience-api

[8] SCORM Overview. (n.d.). Retrieved 26 August 2018, from http://www.adlnet.gov/SCORM

[9] Papadokostaki, K., Panagiotakis, S., Vassilakis, K., and Malamos, A. (2017). Implementing an Adaptive Learning System with the Use of Experience API. In *Interactivity, Game Creation, Design, Learning, and Innovation* (pp. 393–402). Springer.

[10] Moisa, V. (2013). Adaptive learning management system. *Journal of Mobile, Embedded and Distributed Systems*, ***5***(2): 70-77.

[11] Ruano, I., Cano, P., Gámez, J., and Gómez, J. (2016). Advanced LMS integration of SCORM Web laboratories. *IEEE Access*, *4*, 6352-6363.

[12] Lim, K. C. (2015). Case Studies of xAPI Applications to E-Learning. In *The Twelfth International Conference on eLearning for Knowledge-Based Society* (pp. 3.1-3.12.).

[13] 'The LETSI SCORM 2.0 White Papers. (n.d.). Retrieved from https://scorm.com/tincanoverview/the-letsi-scorm-2-0-white-papers/

[14] Experience API, Appendix A: Revision History. (n.d.). Retrieved 26 August 2018, from https://github.com/adlnet/xAPI-Spec/blob/master/xAPI-About.md#Appendix1A

[15] xAPI-Spec. (n.d.). Retrieved 25 May 2018, from https://github.com/adlnet/xAPI-Spec/

[16] Glahn, C. (2013). Using the ADL experience API for mobile learning, sensing, informing, encouraging, orchestrating. In *proceedings of 7th International Conference on Next Generation Mobile Applications, Services, and Technologies* (pp. 268–273). http://doi.org/10.1109/NGMAST.2013.55

[17] Tin Can API Registry. (n.d.). Retrieved 26 August 2018, from https://registry.tincanapi.com/#home

[18] Benedek, A. (2013). Learning Design Versus Learning Experience Design: Is the Experience Api Making the Difference? In *proceedings of Edulearn13: 5th International Conference on Education and New Learning Technologies* (pp. 2609–2621).

[19] Del Blanco, Á., Serrano, Á., Freire, M., Martínez-Ortiz, I., and Fernández-Manjón, B. (2013, March). E-Learning standards and learning analytics. Can data collection be improved by using standard data models?. In *proceedings of Global Engineering Education Conference (EDUCON), 2013 IEEE* (pp. 1255-1261). IEEE.

[20] Paramythis, A., and Loidl-reisinger, S. (2003). Adaptive Learning Environments and e-Learning Standards. In *proceedings of Second european conference on e-learning*, (pp. 369–379).

[21] Poeppelman, T. R., Hruska, M., Ayers, J., Long, R., Amburn, C., and Bink, M. (2013). Interoperable performance assessment using the Experience API. In *Proceedings of the Interservice/Industry Training, Simulation and Education Conference*.

[22] Del Blanco, Á., Serrano, Á., Freire, M., Martínez-Ortiz, I., and Fernández-Manjón, B. (2013). E-Learning standards and learning analytics. Can data collection be improved by using standard data models?. In *proceedings of Global Engineering Education Conference (EDUCON), 2013 IEEE* (pp. 1255-1261). IEEE.

[23] Siemens, G., and Long, P. (2011). Penetrating the Fog: Analytics in Learning and Education. *EDUCAUSE Review*, ***46***(5), 30.

[24] Nine Practical Uses of the Tin Can API (xAPI). (n.d.). Retrieved 26 August 2018, from https://www.youtube.com/watch?v=8LFhDdqQ13A

[25] Corbi, A., and Burgos, D. (2014). Review of current student-monitoring techniques used in elearning-focused recommender systems and learning analytics . The Experience API and LIME model case study. *IJIMAI*, ***2***(7): 44–52. http://doi.org/10.9781/ijimai.2014.276

[26] Burgos, D. (2013). LIME A recommendation model for informal and formal learning, engaged. *International Journal of Interactive Multimedia & Artificial Intelligence*, ***2***(2).

[27] Zhao, X., and Okamoto, T. (2011). Adaptive multimedia content delivery for context-aware u-learning. *International Journal of Mobile Learning and Organisation*, ***5***(1) : 46-63.

[28] Jeong, H. Y., and Yi, G. (2014). A service based adaptive u-learning system using UX. *The Scientific World Journal*, *2014*.

[29] Chen, C. C., and Huang, T. C. (2012). Learning in a u-Museum: Developing a context-aware ubiquitous learning environment. *Computers & Education*, ***59***(3):873-883.

[30] SHIH, S. C., KUO, B. C., and LIU, Y. L. (2012). Adaptively ubiquitous learning in campus math path. *Educational Technology & Society*, ***15***(2):298-308.

[31] ADL's Open Source Learning Record Store (LRS). (n.d.). Retrieved 25 May 2018, from https://github.com/adlnet/ADL_LRS

[32] Articulate Storyline. (n.d.). Retrieved 26 August 2018, from https://articulate.com/p/downloads

[33] TinCan Javascript Library. (n.d.). Retrieved 4 August 2018, from http://rusticisoftware.github.io/TinCanJS/

[34] xapi-statement-viewer. (n.d.). Retrieved 25 May 2018, from https://github.com/adlnet/xapi-statement-viewer

[35] adlnet/xAPIWrapper: Wrapper to simplify communication to an LRS. (n.d.). Retrieved 25 May 2018, from https://github.com/adlnet/xAPIWrapper

[36] xAPI Dashboard. (n.d.). Retrieved 25 May 2018, from https://github.com/adlnet/xAPI-Dashboard/blob/master/API_dashboard.md

[37] Experience API Adopters. (n.d.). Retrieved 26 August 2018, from https://xapi.com/adopters/

[38] POULAKAKIS, Y., VASSILAKIS, K., KALOGIANNAKIS, M., and PANAGIOTAKIS, S. (2017). Ontological modeling of educational resources: a proposed implementation for Greek schools. *Education and Information Technologies*, ***22***(4): 1737–1755.

[39] POULAKAKIS, Y., VASSILAKIS, K., KALOGIANNAKIS, M., and PANAGIOTAKIS, S. (2015). Metadata application profile for primary and secondary education in Greece. In *proceedings of 8th International Conference in Open and Distance Learning (ICODL)* (pp. 45–54). Athens, Greece