Case Studies of xAPI Applications to E-Learning

Kin Chew Lim
SIM University (UniSIM), Singapore
kclim@unisim.edu.sg

Abstract - **The purpose of this paper is to demonstrate the applications of the Experience API (or xAPI – Experience Application Programming Interface, and also known as Tin Can API) in a few e-learning examples. The xAPI is a new specification for learning technology. It allows one to capture data in a consistent format about a person or group's activities from many technologies. Different systems can communicate securely by capturing and sharing the activity streams using xAPI's simple vocabulary. xAPI is regarded as the next evolution to SCORM (Sharable Content Object Reference Model). SCORM is used for packaging e-learning content for interoperability of LMSs. However, SCORM is now considered obsolete. xAPI is an open source API. This allows software programs to read and write experiential data in terms of statements like "I did this", or "actor verb object". Learning activities like "I attended Conference C", or "I tweeted Tweet E to Twitter" are stored in the LRS (Learning Record Store).**

The first case study involves tracking in the game (Oregon Trail). Tracking is useful in order to spot trends and make judgments about what activities are actually working to help people learn things. This case study shows how the various game activities are translated to the xAPI statements. These can then be automatically recorded in the LRS. If the data can co-exist with other learning data (e.g. test results) then we can understand how the student learn to play in the game. The second case study is the LIME (Learning, Interaction, Mentoring and Evaluation) model case study. The authors, Corbi and Burgos (Corbi & Burgos, 2014), explained how the xAPI and LIME model are used in their study which helped them to monitor their students and make recommendations in their studies. In this case, the xAPI LRS is used as an e-learning monitoring engine.

Since the xAPI specifications were first released in 2013, they have received widespread industry acceptance. It is also expected that there will be many more xAPI applications for e-learning.

Keywords - **Activity Stream, Game Learning, Recommender System, xAPI**

I. INTRODUCTION

When computers were first invented in the late 1940s and early 1950s, they were developed and used mainly for the military, government and large corporate users. It was only in 1960 that the first computer- based training (CBT) program was introduced [1]. This was the PLATO, or Programmed Logic for Automated Teaching Operations [2]. It was originally designed for the University of Illinois students but ended up being used in schools throughout the area [3]. Technology-based training (TBT) and teaching using technology accelerated after personal computers were introduced by IBM in the early 1980s.

Subsequently, many courseware titles were developed and delivered via CD-ROMs and laser disks. These gave way gradually to the learning management systems (LMSs) when computer systems became more powerful and could store more contents. It was the AICC [4] which released the first specification for

the LMS. This specification allowed students' scores to be tracked on the computer system he was using. When the Internet became a world-wide sensation in mid-1990s, Web-Based Training (WBT), virtual classrooms and e-learning in general became fashionable. At about the same time, several learning standards consortia were founded. These included the IMS Global Learning Consortium [5] and the Advanced Distributed Learning (ADL) Project [6].

SCORM (Shareable Content Object Reference Model) was released by the ADL Project in the year 2000.

SCORM is the de facto specification for packaging learning content is a standard format which allows the package to work in different LMSs. However, SCORM is tied very closely with the LMS. It will not work outside of the LMS and the browser.

A. Shortcomings of Present LMS-Centric and Content-Centric E-Learning

So far, the approach in e-learning is to deal with how the content is to be structured, packaged and moved from one system to another. This is a very LMS- and content-centric model. SCORM is thus very LMS- and content-centric and hence it has many restrictions.

For example, multiple-choice quizzes are used widely in the LMS-centric model. These quizzes are usually of the single- answer assessments. Questions with single answers do not reflect real world situations in which there might not be single-solution answers. Learners also could guess the answers. The materials provided in the LMS are mostly textual in nature although occasional video and animation clips were used. The LMS-centric model will always have the teacher as the knowledge dispenser. Participants in an LMS-centric model do not share much. In addition, contents from other devices (e.g. smartphones, tablets and social media) were difficult to be consolidated with those on the LMS. The LMS must be connected to the Internet all the time in order for learning interactions to take place. On the other hand, smartphones are not always connected to the Internet. Finally, it is difficult to ascertain how much learning the participant has done if he or she uses multiple devices to access information [7].

B. What led to the Development of the xAPI?

SCORM was first released in 2000 [8]. It has served its purpose of achieving interoperability in different LMSs. But since then the landscape has changed tremendously. Firstly, there is an extensive worldwide proliferation of mobile devices and the mobile app ecosystem. People are now using different mobile devices to receive information, communicating, learning and collaborating amongst themselves. At the same time wireless and Wi-Fi coverage are increasing everywhere. People everywhere engaged in games, whether on the web, using the console or mobile devices. Applications in augmented reality and simulations are spreading not only on the desktop computers but on mobile devices like the iPADs. People are also communicating extensively using social media tools like Facebook, Twitter, Instagram and blogs. Open source movement is gaining widespread use with people everywhere [9].

A person might be texting one moment. Next moment, he used a desktop computer to access an LMS to do an online quiz. After a while he might be in a restaurant discussing business deals with his client. For this, he used an iPAD. Later in the afternoon, he could be attending a 1-hour webinar using his Android smartphone. All these activities show that very little online learning happens on the LMS! The LMS is used only as a repository of learning materials.

Subsequently, the ADL of the US Department of Defense engaged Rustici, an e-learning software company, to work on a new proposal for the new generation of e-learning specification. After extensive consultations with the e-learning community, Rustici developed the Tin Can API in 2013. The ADL later renamed it xAPI, for Experience API. Version 1 of this

specification was released in April 2013 [10]. The current version is at version 1.0.2 [11]. Your paper must be in two column format with a space of 4.2mm (0.2") between columns.

II. WHAT IS THE XAPI?

The Experience API forms part of the Training and Learning Architecture (TLA) [6] that the Advanced Distributed Learning (ADL) Project is working on. This API (also known as the Tin Can API), is an open source e-learning software specification [12]. The specification makes it possible to collect data about the learning experiences a person has achieved either online or offline. Learning experiences are recorded in a Learning Record Store (LRS). LRSs can exist within traditional Learning Management Systems (LMSs) or on their own [13].

The Experience API is commonly considered the successor to SCORM (Sharable Content Object Reference Model) [14]. Since 2000, SCORM has been the de facto e-learning standard for packaging e-learning content to be delivered to LMSs. (Training Industry Magazine, 2014). However, there are several drawbacks to SCORM [15].

This API is stewarded by ADL. xAPI focuses on how the activities people do are evidence of a learning experience. It is a Representational state transfer (REST) web service. As for the data format, it uses the JavaScript Object Notation (JSON). The web service allows software clients to read and write experiential data in the form of "statement" objects. Statements are in the form of "I did this", or more generally "actor verb object". [16]. More complex statement forms can be used.

With the xAPI, e-learners can take e-learning outside of the browser [17]. In addition, xAPI allows e-learning to execute in native mobile applications [18]. Thus, there is more control over the learning content should the xAPI specification be used. Not only that, there is better security using a technology called Oauth [19]. Another use of the xAPI is that of platform transition; e.g. an e-learner starts e-learning on a mobile device and finishes it on a computer [19]. Other possibilities include those of tracking games and simulations [13], tracking real-world performance [20], tracking team-based e-learning [13] and tracking learning plans and goals [21].

A. xAPI Statements

The xAPI is a web service. A web service supports applications on the World Wide Web (WWW) and makes use of the HyperText Transfer Protocol (HTTP). As a web service, the xAPI allows for statements of experience, typically learning experiences, to be delivered to and stored securely in a Learning Record Store (LRS).

The web service allows clients to read and write experiential data in the form of "statement" objects. In their simplest form, statements take the form of "I did this", or more generally "actor verb object". xAPI also provides facilities for more complex statement forms [22].

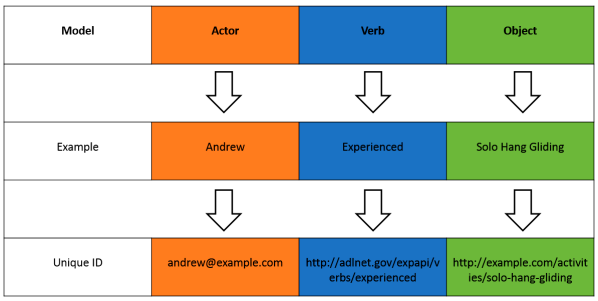

Fig 1. The Basic Elements and Structure of an xAPI Statement

In the example, "Andrew experienced 'Solo Hang Gliding'", we recognize that "Andrew" is the actor, "experienced" is the verb, and "Solo Hang Gliding" is the activity. The statement object itself would take this structure in JSON (JavaScript Object Notation) format:

```
    "actor": "Andrew", "verb":
    "experienced",
    "object": "Solo Hang Gliding"
}
```

This is a simple example. How do we know which Andrew we mean? Which 'Solo Hang Gliding' activity was it? Was it one that was part of military training, or the one from a commercial enterprise, or something self-directed? Here is a valid xAPI statement:

```
{
   "actor": {
      "name": "Andrew Downes",
   "mbox":  mailto:andrew@example.com
},
   "verb": {
      "id":
"http://adlnet.gov/expapi/verbs/experienced",
   "display": {"en-US": "experienced"}
      },
   "object": {
      "id":
"http://example.com/activities/solo-hang-
gliding",
      "definition": {
         "name": { "en-US": "Solo Hang
Gliding" }
   }
  }
}
```

A structure has been added here to ensure that we can uniquely identify the component parts. This helps to correlate statements about the same person, activity, or verb. There is also a structure added to provide information about the objects, like name. Other descriptive fields are available (Experience API, 2015).

We can also add a lot more to a statement, in the form of statement context ("Andrew completed 'Solo Hang Gliding' in the context of 'Army Training Level 1'" or "Bob completed 'Truck Driving Training Level 1' on his Android phone, under the instruction of Dan") or you can attach results to a statement ("Bob passed 'Truck Driving Training' with score 90%"). You can even declare custom fields on a statement, in the form of extensions (Rustici Software 4, n.d.).

B. Learning Record Store (LRS)

A Learning Record Store (LRS) is a place to store learning records. The LRS is a new system that goes together with the xAPI. As xAPI-enabled activities generate statements, they are sent to an LRS. The LRS is simply a repository for learning records that can be accessed by an LMS or a reporting tool. An LRS can live inside an LMS, or it can stand on its own. LRSs record all of the statements made. An LRS can share these statements with other LRSs.

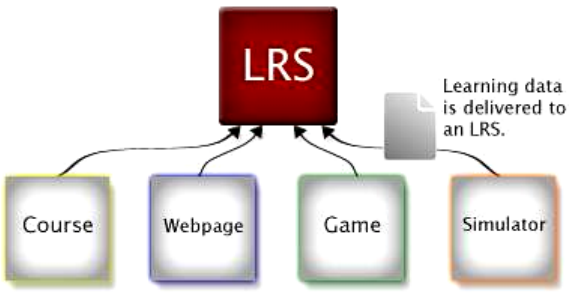

Fig 2. Learning Record Store (LRS)

The data stored in an LRS can be accessed by LMSs, reporting tools, or other LRSs. The data can be stored as individual learning records and/or entire transcripts. An LRS can limit who can read and write learning records.

SCORM and other e-learning standards only store a certain amount of learning data. xAPI allows for the LRS to store nearly everything. This means better reporting and a much more accurate picture of learners.

An LRS can live in an LMS and use the LMS's reporting tools to make meaning of the LRS's data. Or it can live on its own with its own reporting tools.

LRSs can share data amongst themselves, so learners and data can be transferred from one organization to another. Statements can also be sent to multiple LRSs (e.g. "I want to record my training in my own personal LRS as well as my employer's LRS.")

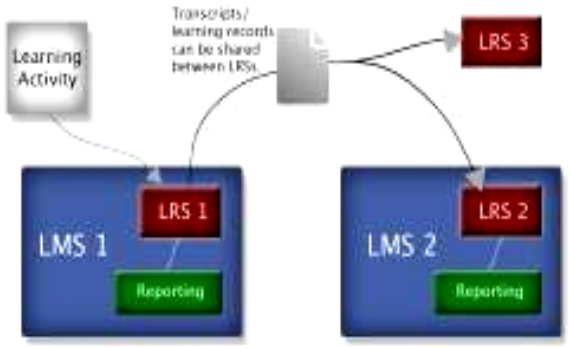

Fig 3. Sharing Data among LRSs and LMSs

III. CASE STUDIES IN ELEARNING

A. Oregon Trail Game

Background - The Oregon Trail is a computer game [23]. It was originally developed by Don Rawitsch, Bill Heinemann, and Paul Dillenberger in 1971. It was produced by the Minnesota Educational Computing Consortium (MECC) in 1974. This game was designed to teach school children about the realities of 19th century pioneer life on the Oregon Trail. The player assumes the role of a wagon leader guiding his or her party of settlers from Independence, Missouri, to Oregon's Willamette Valley on the Oregon Trail via a covered wagon in 1848. The game is the first entry in the Oregon Trail series of games. It has since been released in many editions by various developers and publishers. The Oregon Trail was extremely successful. It sold over 65 million copies [24].

The Details - Oregon Trail is an e-learning game. First developed in 1971, it has since then been played on many platforms such as Apple computers, Windows computers, iOS devices, Nintendo, and even Blackberrys. In fact, it has been sold 65 million copies.

As an e-learning game, it teaches people by making learning fun and interactive for the learner. Unfortunately, most of the learning activities that are created do not get recorded. These learning activities create valuable learning experiences for the learner. But they are often not measured.

Older e-learning standards did not work for recording the experiences created in the game. The game do not always get played in a browser. It is not always being played on a "connected" device. It was definitely not played in an LMS. However, sometimes it is played as a mobile app. These things do not work with older standards like SCORM and AICC.

Why would anyone want to record the learning experiences created in the game? Furthermore, why would we use an e- learning standard to record the learning experiences? Well, just like any other learning data, tracking it is useful to spot trends and make judgments about what activities are actually working to help people learn things. If this data could be tracked such that it could live side by side with other learning data (test results, for example) then the big picture of teaching and learning is easy to see.

The xAPI (or Tin Can API) makes the game much more useful. Firstly, it makes it possible to record all the learning data that is generated. Secondly, that data lives side by side with all of the other data that being generated by students and teachers.

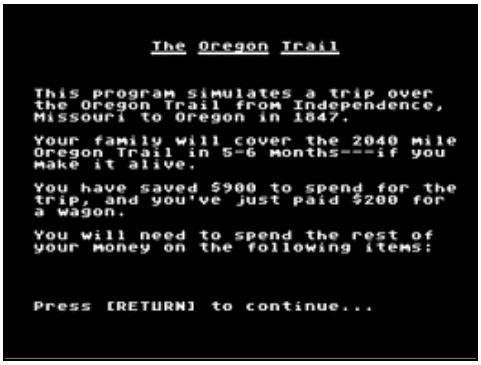

Fig 4. The Oregon Trail Game

Fig 5. Oregon Trail – e.g. Alan Choose to be a Carpenter

Fig 6. Oregon Trail – "You have Crossed Kansas River"

The xAPI statements, as played by the character, Alan, are as follows:

Alan's Experience	xAPI Statement
Alan chose to be a carpenter.	Alan completed "choose occupation" with a result of "carpenter."
Alan added Ryan to his party.	Alan completed "add member to your party" with result of Ryan.
Alan added Mike to his party.	Alan completed "add member to your party" with result of Mike.
Nicole got dysentery, and Alan chose to continue his journey rather than take action.	Alan experienced "party member getting dysentery." Alan experienced "ignore dysentery and continue on journey."
Alan made it to the Kansas River.	Alan completed "arrival at Kansas River crossing".

There are two ways to track the learning in the Oregon Trail game. The first way is to access the game's program codes and incorporate the xAPI statements. However, this might not be possible if the game's codes are not accessible to other people. The second way is to develop the simple program which allows the various xAPI statements to be recorded directly into the LRS. This is the preferred way.

Games are an immersive and effective way for people to learn, but they have previously not been well tracked. xAPI allows one to track whatever one wants to in a game. It also allows the learning data to exist in the same system as all of the other learning data of the learner. Instructors can see the big picture of learning and teaching. It also helps them to better figure out effective ways to teach.

B. LIME Model Case Study

The second case study is the LIME (Learning, Interaction, Mentoring and Evaluation) model case study. According to the authors, Corbi and Burgos [25], this case study helped them to monitor their students and make recommendations in their studies.

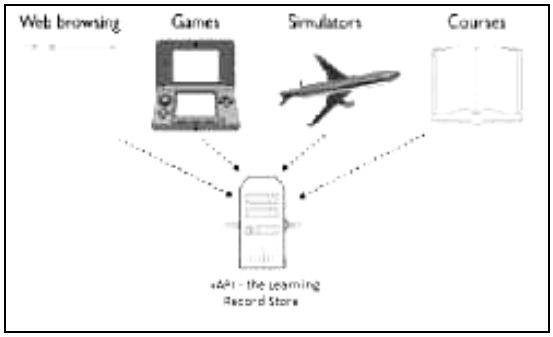

Fig 7. Examples of using the xAPI

1. ***Recommender Engines*:** In recent years, recommender engines have become very popular. You can now find recommendation engines in books, movies, music, news, products, research articles, search queries, and social tags in general. These engines deliver suggestions based on the collected information on preferences, general user behavior and even items bought or content searched. Students depend on recommendations from their peers and professors in order to do their research. Lately, the research community is paying much attention to recommender engines.

Fig. 8 shows, on the Y-axis, the number of cited papers from each year as of 2013. There is a peak of interest around the year 2009.

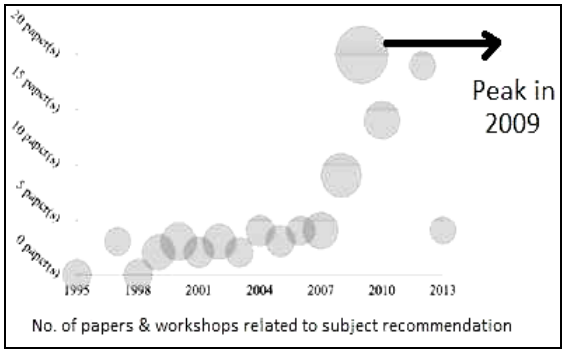

Fig 8. No. of Papers & Workshops Versus Year

Corbi and Burgos (2014) proposed using the LIME model [26] to develop the recommender engine. The LIME model is based on four vectors:

i. What every learner does based on his/her own contribution (L = Learning)
ii. What the learner does to support interaction-based learning and the relation with others, in addition to group interaction (I = Interaction)
iii. What teachers/experts value (M=Mentoring)
iv. A transversal vector is applied to the three previous vectors, focused on evaluation (E=Evaluation)

Fig 9. Examples of the LIME Model

2. ***xAPI as an E-Learning Monitoring Engine:*** The LRS (Learning Record Store) is the core of the xAPI. The LRS is a specific module for data storage that allows an LMS to report tracking information on the learning experience. At any time, an LMS can send collected data over the network to an xAPI web service. An LRS is a wrapper or API software layer to a SQL database. This free LRS implementation was open-sourced by ADL (Advanced Distributed Learning). It is based on the Python computer language and on the Django web framework.

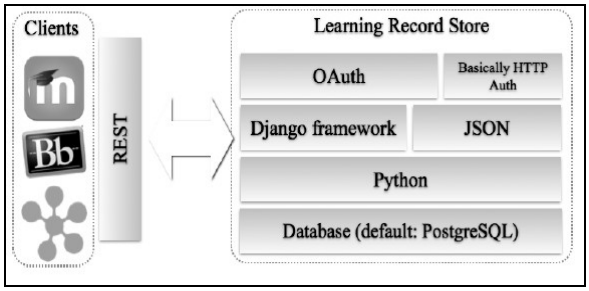

Fig 10. LRS Software Stack & Interaction

When students interact with educational content using different systems or tools, they will leave traces in the LRS. Each tool will provide a totally different actor/user ID to ensure anonymity.

The *verb* element is a key part of an LRS communication. This is because it describes the action performed by the student. In an elearning environment, a verb could be something like: "write", "read", "experimented", "passed", "failed", "experienced", etc.

The *object/activity* part of the statement refers to "what" was experienced in the action defined in the verb. It usually corresponds to the learning activity (e.g. twitter, webinar, wiki, chat room, forum, mail message, etc.). Objects/activities must also embody a URL (Universal Resource Locator) pointing to their rationale. This can include other information such as the learning activity's description, verbs that can apply, possible results and usage suggestions. The *result* component provides the outcome to the statement. It includes score, level of success and completion fields.

The context part adds more details to the overall statement, like the relationship of the activity with other activities, its order in the learning stream, or the teacher's name.

An LRS must also implement REST calls for data transfer (PUT, POST, GET and DELETE). The xAPI can make use of either

OAuth or HTTP Basic Authentication when communicating securely with the outside world.

One key aspect of the LRS architecture is that it can be implemented in shared cloud ecosystems. This allows communications from very different elearning platforms and academic institutions. In other words, monitoring data can be uniformly stored. This allows rapid, vast and democratic access to learning analytics information. There are some free LRS *hosting* services but mainly for testing and technology promotion purposes. One such service is by the ADL (http://lrs.adlnet.gov/xAPI) and another one by Rustici Software (http://tincanapi.com/prototypes/).

3. ***The LIME Model and the LRS:*** Lecturers-tutors must design a strategy for each of his/her courses. The model codifies this strategy for a course or class group by using settings and categories.

A course setting is the balance between formal and informal scenarios. In this context, *formal* means a regular academic program with regular evaluation means (e.g. graded exams); *informal* means continuous evaluation and user activity inside the LMS and every tool linked to it (e.g. Social Networks or repository). The system collects specific inputs from both settings, keeping an overall balance of 100%. For instance, if the designer requires just a *formal* setting, the balance should be *Informal*: 100% - *Formal*: 0%.

Furthermore, a learning scenario must be defined as the balance between the Learning, Interaction, Mentoring, and Evaluation, in combination with the Formal and Informal settings categories. In the LIME model, every category and setting are assigned with a specific weight (*wi*), keeping an overall balance of 100%.

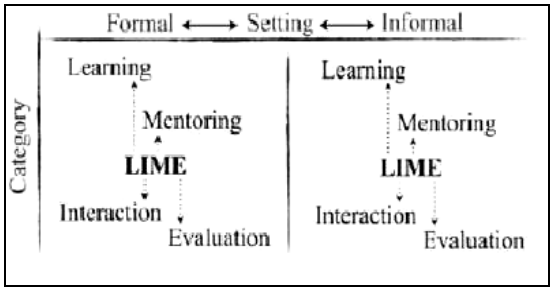

Fig 11. Categories & Settings in the LIME Model

In the LIME model each input (action performed by a student in the eLearning platform or Social Network) is attributed a category and a weight, assigned by the instructor. An example of model configuration for a specific site can be found in Fig. 12. Based on these components, the lecturers-tutors can manually define and parameterize recommendation rules. These rules will only trigger a message to the student if conditions regarding categories, inputs and settings are met.

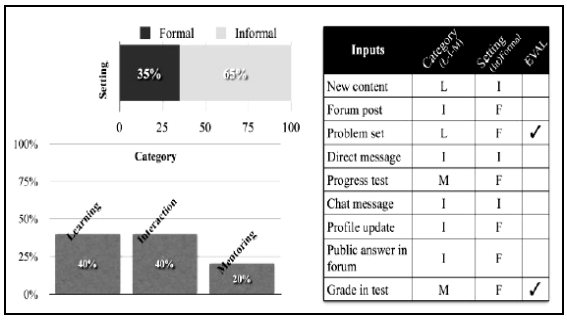

Fig 12. Sample Configuration of the LIME Model for a Specific Course Site

LIME is a lecturer-tutor-crafted, rule-based recommender system for learning environments on the cloud. LIME's goal is simply to improve learning efficiency, and to facilitate the learning journey of every student by a personalized recommendation set. LIME can be fed from learner inputs in many ways. However, this model can also be initialized with tracked data stored in an xAPI LRS instance/server.

How can LIME inputs be built out of information stored in the LRS? A LIME model input has to define an action and a context in which a learner performs this action:

- participate in chat
- answer in main forum thread
- message to tutor
- resolve a problem set
- formally broadcast mail to mates

xAPI verbs and objects, taken in an isolated way, are not sufficient. However, a joint entity composed of a verb plus an xAPI object makes more sense.

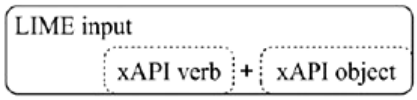

Fig 13. LIME Inputs from xAPI Statements

It is up to the implementer to define which verbs and objects best represent the scenario to be tracked and monitored. Let us take a look at the sample verbs and activities available on the official xAPI site (i.e. http://adlnet.gov/expapi).

In Fig. 15 are listed all the verbs and activities the LRS can store. We also have their possible combinations to build a meaningful and compatible LIME input.

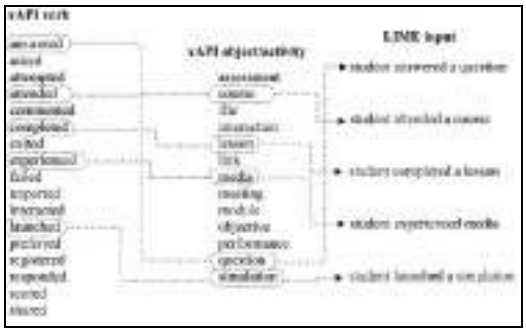

Fig 14. xAPI Verbs and Objects to LIME Inputs

Each input should be assigned a weight (*wi*), a category and a setting. These parameters should reside on the LIME system's own configuration repository. In other words, LIME administrators should maintain an updated *equivalency list* between LRS vocabulary and LIME inputs. These inputs will then interplay with rules (Figure 11), which are, in turn, based on *predicate filtering*.

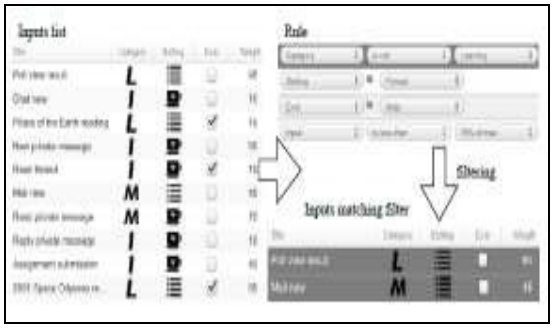

Fig 15. Predicate Filtering in LIME

As LIME was developed as a Basic Learning Tool Interoperability (Basic LTI) [27] application, this equivalency list can even be stored in the LMS database through the IMS LTI Settings API specification, part of IMS LTI 1.0 and above. The LIME model thus remains free from external configuration files or own database management.

It is important to notice that LMS must be LTI compatible and support the Settings API protocol.

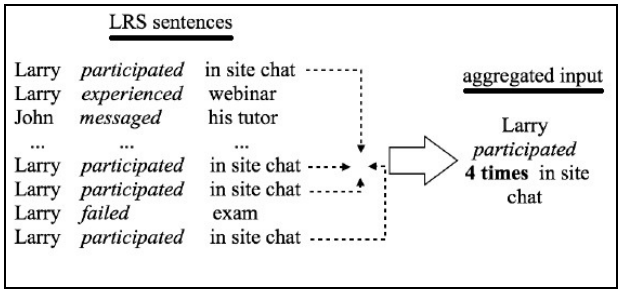

Fig 16. Aggregation of LRS Sentences

Mathematically:

$$(\text{LIME input})_i = \frac{\sum_{j=0}^{n} (\text{LRS statement})_j}{n_i}$$

These aggregation operations are covered by the xAPI standard. The xAPI provides a query language to easily data-mine an LRS. For instance, the following code collects all the times the user "Alan" has tried an exam, and returns an aggregated result:

```
stmts.where (
'actor.name = "Alan" and ('+
'verb.id =
"http://adlnet.gov/expapi/verbs/passed"'+
'or '+
```

'verb.id = "http://adlnet.gov/expapi/verbs/failed"'+ ')')

IV. CONCLUSION

This paper traces the background of the Experience API, or xAPI in short.

Where previously much attention was placed on structuring the e-learning contents, the trend is now moving towards measuring learning experiences. This comes about as there is realization that new technologies like mobile, virtual and multimedia technologies are constantly changing. However, it is learning that all of us are more interested in.

xAPI has been designed to store use data in a simple, centric, standard, client agnostic and powerful way. A central part of this is the Learning Record Store, or LRS in short. With this record store, one can measure the learning component in any elearning activity.

The first case study is about measuring the learning that can take place in the Oregon Trail game. This is an adventure game in which a person can take on a role and goes on an adventure trail called the Oregon Trail. The gamer can assume various roles in his journey across the USA. He can cross a river, eat some food and even fall sick during his journal. xAPI statements can be drafted and stored as indications of his or her learning experiences. These statements can be stored on a Learning Record Store (LRS). The second case study is about using the xAPI statements to make recommendations for the student. Recommender engines or systems are becoming popular in many activities like ordering books, reading research papers, booking hotels, going to restaurants, choosing movies, buying cars and making investments. The core of the second case study is the LIME (Learning, Interaction, Mentoring and Evaluation) model. With this model, it is possible to capture the formal and informal learning processes going on. By using the LIME model, every category and setting assigned with a specific weight (wi). An overall balance of 100% is kept (i.e. formal learning+ informal learning = 100%). The lecturers- tutors can then manually define and parameterize recommendation rules. These rules will only trigger a message to the student if conditions regarding categories, inputs and settings are met.

These two case studies are just some of the increasing number of xAPI applications being launched rapidly. The author feels that with widespread support from the industry, government and academic bodies, there will be even more and better xAPI applications in the future.

REFERENCES

(Arranged in the order of citation in the same fashion as the case of Footnotes.)

[1] Bersin, J. (2004). "The Blended Learning Book: Best practices, proven methodologies and lessons learned". Pfeiffer.

[2] Lombart, P. (2011). "PLATO (Programmed Logic for Automatic Teaching Operations)". <http://whatis.techtarget.com/definition/PLATO-Programmed-Logic-for-Automatic-Teaching-Operations>. Accessed 11 May 2015.

[3] Epignosis, LLC. (2014). "E-Learning concepts, trends, applications". <http://www.talentlms.com/elearning/elearning- 101-jan2014-v1.1.pdf>. Accessed 20 April 2015.

[4] Aviation Industry CBT Committee (AICC). (n.d.). "Aviation Industry Computer-Based Training Committee". <https://en.wikipedia.org/wiki/Aviation_Industry_ Computer-Based_Training_Committee>. Accessed 27 November 2015.

[5] IMS Global Learning Consortium. (2015). "About IMS Global Learning Consortium". <http://www.imsglobal.org/background.html>. Accessed 11 May 2015.

[6] Advanced Distributed Learning. (n.d.). "Advanced Distributed Learning: Capabilities: Training and Learning

Architecture". <http://www.adlnet.gov/index.html>. Accessed 11 May 2015.

[7] Advanced Distributed Learning (ADL) Co-Laboratories. (2012). "An ADL Perspective on Next Generation SCORM Requirements as Derived from Project Tin Can". <http://www.adlnet.gov/wp-content/uploads/2012/01/NEXTGEN-SCORM- requirements-20120130_v1.pdf>. Accessed 10 May 2015.

[8] Glahn, C. (2013). "TinCan and The Confusion About the Next Generation of SCORM". <http://lo-f.at/glahn/2013/05/the-confusion-about-the-next-generation-of-scorm.html>. Accessed 10 May 2015.

[9] Hruska, N. (2013). "The Experience API". <http://www.adlnet.org/wp-content/uploads/2013/04/The_Experience_API_i n_Practice.pdf>. Accessed 21 April 2015.

[10] GitHub: Experience API. (2015). "GitHub Web site". <https://github.com/adlnet/xAPI-Spec/blob/1.0.3/xAPI.md>. Accessed 9 May 2015.

[11] Experience API. "At version 1.0.2. (2015)". <https://github.com/adlnet/xAPI-Spec/blob/master/xAPI.md>. Accessed 27 November 2015.

[12] Bowe, M. (2013). "The Open Source Landscape". <http://tincanapi.com/2013/07/11/the-open-source- landscape/>. Accessed 19 April 2015.

[13] Brusino, J. (2012). "The next generation of SCORM: a Q&A with Aaron Silvers". American Society for Training and Development. <https://www.td.org/Publications/Newsletters/Lear ning-Circuits/Learning-Circuits- Archives/2012/06/The-Next-Generation-of-SCORM-a-Q-and-a-with-Aaron-Silvers>. Accessed 19 April 2015.

[14] Tillett, J. (2012). "Project Tin Can – The Next Generation of SCORM". Project Tin Can – The Next Generation of SCORM. Float Mobile Learning. <http://floatlearning.com/2012/04/project-tin-can- the-next-generation-of-scorm/>. Accessed 19 April 2015.

[15] Whitaker, A. (2012). "An Introduction to the Tin Can API". An Introduction to the Tin Can API. The Training Business. <http://www.thetrainingbusiness.com/softwaretool s/tin-can-api>. Accessed 19 April 2015.

[16] Tillett, J. (2012). "Saltbox Developers Discuss Tin Can". <http://floatlearning.com/2012/07/saltbox-developers-discuss-tin-can/>. Accessed 10 May 2015.

[17] eLogic Learning. (2012). "eLogic Learning Partners with Rustici Software to be an Early Adopter of the Next Generation of SCORM Standards Known as the 'Tin Can API. (2012)". <http://www.prweb.com/releases/SCORM/e-learning/prweb9610860.htm>. Accessed 19 April 2015.

[18] Brandon, B. (2012). "Making History: mLearnCon 2012 Rocks Attendees". Making History: mLearnCon 2012 Rocks Attendees. Learning Solutions Magazine. <http://www.learningsolutionsmag.com/articles/95 8/>. Accessed 19 April 2015.

[19] Project Tin Can Phase 3 - the future of e-learning is now. (2012). "We Need Security/Authentication". <http://scorm.com/project-tin-can-phase-3-we-need-securityauthentication/>. Accessed 19 April 2015.

[20] Gautam, A. (2012). "Tin Can: My First Impressions From mLearnCon 2012". Tin Can: My First Impressions From mLearnCon 2012. Upside Learning. <http://www.upsidelearning.com/blog/index.php/2 012/06/21/tin-can-my-first-impressions-from-mlearncon-2012/>. Accessed 19 April 2015.

[21] Downes, A. (2012). "I Want This: Tin Can Plans, Goals and Targets". <http://tincanapi.co.uk/pages/I_Want_This.html>. Accessed 19 April 2015.

[22] Rustici Software 4. (n.d.). "Tin Can API:

Statements 101". <http://tincanapi.com/statements-101/> Accessed 10 May 2015.

[23] Oregon Trail. (2014). <https://archive.org/details/msdos_Oregon_Trail_ The_1990>. Accessed 27 November 2015.

[24] Rustici Software. "A Game's Story". <https://tincanapi.com/a-serious-games-story/>. Accessed 27 November 2015.

[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 & LIME model case study. International Journal of Artificial Intelligence and Interactive Multimedia, Vol. 2, No 7. <http://www.ijimai.org/JOURNAL/sites/default/files/files/2014/09/ijimai20142_7_6_pdf_27449.pdf>. Accessed 8 May 2015.

[26] Burgos, D. (2013). "L.I.M.E. A recommendation model for informal and formal learning, engaged". <http://www.ijimai.org/journal/sites/default/files/files/2013/06/ijimai20132_2_11_pdf_25682.pdf>. Accessed 27 November 2015.

[27] Learning Tools Interoperability. (2014). <https://www.imsglobal.org/activity/learning- tools- interoperability>. Accessed 27 November 2015.