The 13th International Scientific Conference
eLearning and software for Education
Bucharest, April 27-28, 2017
10.12753/2066-026X-17-000

Towards the enhancement of AGLOs with SCORM and xAPI

Ciprian-Bogdan Chirila
Department of Computer Science and Software Engineering,
University Politehnica Timişoara, V. Pârvan Blv. no. 2, Timişoara, Romania
chirila@cs.upt.ro

***Abstract**: Generative learning objects (GLO) are considered the second generation learning objects. They represent reusable pedagogical patterns instantiable with different content. Auto-generative learning objects (AGLOs) are reusable patterns filled in with generated content obtained by meta-programing and the use of random numbers to enable content diversity. Static content currently present in most of the Learning Management Systems (LMS), MOOCs etc. and there is the time for dynamic content models due to the web technology present in almost all gadgets the student use: laptops, tablets, mobile phones, watches etc. SCORM is a set of interoperability standards for the e-learning objects in the context of LMSs. It is not related to instructional design or pedagogical concerns. Nowadays SCORM is considered obsolete because of the new standards of xAPI a.k.a. Tin Can API. With xAPI the learning experience is stored based on the "I did this" paradigm. So the student will not get a consolidated final mark where you will not be able to see its components but will have individual subject-action-object based records where its progress can be monitored and learning adjusted according to needs. In this context the AGLOs will be able to adopt this standard based on the competence tree model used in their organization. Recipes have been created for the organization and redundancy avoidance reasons of subjects, actions and objects. We will extract the verbs and objects from the competence tree model and to research the potential for the automation of the process. The approach will be experimented on the data structures and algorithms discipline competence tree developed in earlier research projects.*

***Keywords:** learning objects, generative learning objects, SCORM, xAPI*

I. INTRODUCTION

Nowadays student’s behavior seem to converge with the emerging learning trends. Students use very often mobile devices like: tablets, smart phones triggering the development of mobile learning or mLearning. Students often have limited time gaps between their activities enabling the development of micro-learning components [17], gamifications interactive videos etc. On the other hand students tend to stay online, especially on social networks almost the whole day so this enables the development of collaborative learning on these networks. One current development direction for the domain is in the area of MOOCs [24]. E-learning developments are oriented towards the regional continuously growing automotive industry [3,4].

In this sense auto-generative learning objects (AGLO) adhere to the trends and also to the student needs. AGLOs operates in the context of Data Structures E-Learning (DSEL) platform which is a fully-fledged Web 2.0 based application, relying highly on JavaScript [16] for both AGLO models and implementation. The designed learning time-frame for AGLOs allows them to act as micro-learning components. AGLOs have the potential of generating gamifications that use social networks rewarding to enable student motivation [13].

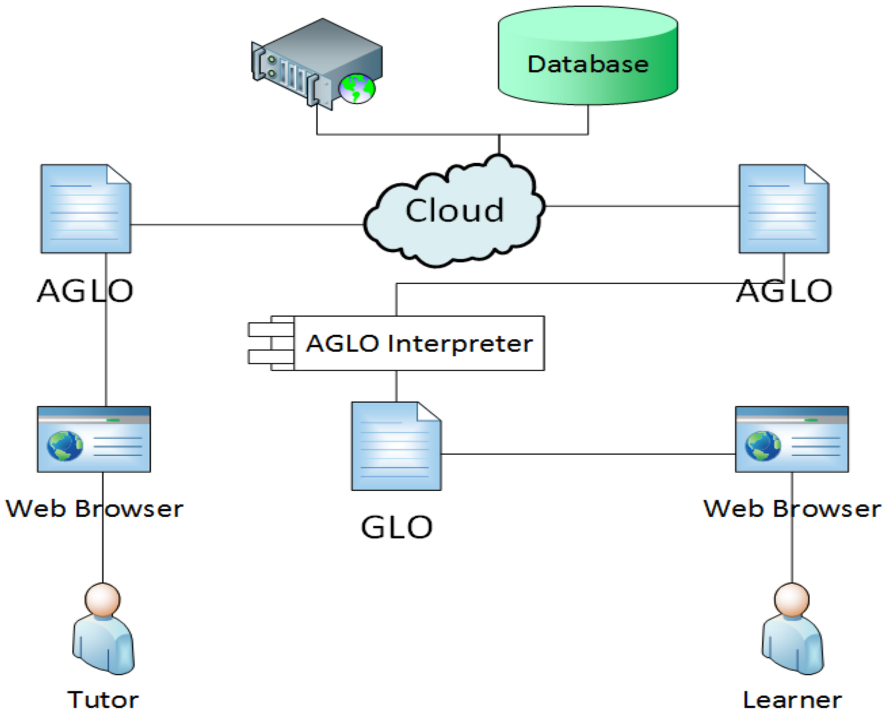

Figure 1 – The AGLO Approach

In Figure 1 we depict the AGLO approach in a cycle starting from the AGLO tutor design and closing with student content consumption and providing feedback to the tutor. The tutor edits online the AGLO model. Depending on the tutors ability the editing may have several levels of complexity: i) to make small changes to the already existing models, e.g.: (1) to adjust parameter values, thus controlling content generation; (2) to translate the content form the original language into a target language; (3) to rephrase some parts of the content, to impose a personal style in the conversation etc.; ii) to change the AGLO model entirely, e.g.: for repurposing it for another related learning objective; iii) to create AGLO models from scratch.

In order to have a better control of the content to be delivered to the student, the tutor has to test the generation result and to evaluate its quality. Specific such aspects are: semantic errors, content layout, etc. that can be corrected by changing the parameter values in the model.

Experience API (xAPI) [1] is a framework for the recording the student actions as statement objects. The objects structure follow the “I did this” paradigm composed of actor, verb and target. It is based on the “ActivityStreams” format created by Google, Facebook, Microsoft and IBM to record user actions and to generate big data.

The paper is structured as follows. In chapter II we present related works. In chapter III we present the model of generative learning objects. Chapter IV presents the structure of the competence tree model and some content examples. In chapter V we present the integration model. Chapter VII concludes and sets the perspectives.

II. RELATED WORKS

The Learning Object Metadata (LOM) standard [20] describes learning objects and other learning resources using XML markup language to enable reusability, discoverability and interoperability in the ecosystem provided by LMSs. Our AGLO model is expressed in XML having additional JavaScript expressions and it is targeted towards reusability based on parameters variation and on patterns reuse. Enabling xAPI statements into AGLO increases the interoperability in the context of LMSs.

The Sharable Content Object Reference Model (SCORM) [2] is a standard developed by the ADL Initiative for facilitating the reuse of learning objects. Thus, the instructional components were separated from their specific run-time constraints and incorporated into different applications. The SCORM standard contains a set of technical specifications and guidelines for creating interoperable, plug and play, web enabled e-learning content. The approach is based on: a generic run-time environment (RTE) with a data model and API for creating e-learning content, enabling standardized communication between LMSs; a content aggregation model (CAM) that explains how to package content and a sequencing and navigation (SN) model. Our AGLO model does not adhere to the SCORM model since it became obsolete and new ideas were embraced by the e-learning community.

The Experience API (xAPI) [1] is a framework that allows for big data capturing about human performance, associated instructional content and performance context information. It can track micro-behaviors, states, contexts. The framework is free, lightweight, adaptable and present in multiple LMSs so this motivated us towards integrating it into our AGLO model.

[5,6,7,21] design a framework for authoring reusable and re-targetable learning objects. In this context generative learning objects (GLO) are reusable pedagogical patterns that can be re-purposed towards different learning objectives. Our AGLO model adheres to this concept and enhances it with metaprogramming and random number generators in generating the learning content.

[14,15,25,26,27] develop a GLO approach based on metaprogramming for teaching programming using Lego robots. Their approach relies on several aspects like: feature models for modeling variability and sequencing, metaprogramming for content generation. Our approach is very similar the previously mentioned ideas but it is more pragmatic in the sense of authoring and implementation being targeted towards computer science programming disciplines like Data Structures and Algorithms.

[18] presents a 3D animation framework that motivates students to learn object-oriented programming in Java by creating games. The framework targets skills like ambition, desire to win, strategic thinking, motivation and perseverance.

[19] presents a framework based on Flex, ActionScript and Flash that allows the configuration of generative learning objects created by a designer and that are integrated in a user defined application scenario controlled by a programmer. Both approaches their and ours rely on XML templates MXML, respectively XML mixed with JavaScript for the metaprogramming. The main difference between approaches is the use of random numbers and the lack of programmer in the scenario creation stage.

III. GENERATIVE LEARNING OBJECTS IN DSEL FRAMEWORK

The AGLO model [12] is composed of six sections (see Figure 2). The first section is trivial and contains the name of the model. Usually the name indicates some of the operations and objects the model is about. For example, we can have a title like: to draw a tree using the parent reference representation. This information can be used further for easily creating statements. The next section is the core of the model and is the scenario where informal descriptions are provided by the designer of the model in natural language like code comments.

In the scenario sections symbols are defined. They have a name and an instantiation expression written in JavaScript and relying on random numbers. Symbols may use in the initialization expression the names of other previously defined symbols. Symbols may have several types like: integer, real, string. They can have simple values but also SVG values for the representation of dynamically created diagrams like trees and graphs. We remind that in order to enable a dynamic content and to create different material at each instantiation it is required that instantiation expressions should call random number generator functions.

The next section is the theory section where theoretical information may be provided. Reading this information the learner must be able to answer the incoming question. The natural format for this section is HTML.

```
01 AGLODef ::= <action> Name Scenario [Theory] Question Answers Feedbacks </action>
02 Name ::= <name> (ID)* </name>
03 Scenario ::= <scenario> [ Comment ] Symbol* </scenario>
04 Comment ::= (ID|CT)*
05 Symbol ::= <symbol SymbolName Type> Expression </symbol>
06 SymbolName ::= name = ID
07 Type ::= type = (boolean | int | float | double | string | array)
08 Expression ::=<expr> Function ( ExpressionList ) </expr>
09 ExpressionList ::= Expression (, Expression)*
10 Function ::= (item from composed functions and operators list of JavaScript
using random numbers)
11 Theory ::= <theory> (ID)* </theory>
12 Question ::= <question> (ID | Value)* </question>
13 Value ::= <value name = ID />
14 Answers ::= <answers> (Answer)+ </answers>
15 Answer ::= <answer id = Index >(ID | Value)* Correctness </answer>
16 Index ::= INTEGER_LITERAL
17 Correctness ::= <correct> (true | false) </correct>
18 Feedbacks ::= <feedbacks> (Feedback)+ </feedbacks>
19 Feedback ::= <feedback> AnswerIdList (ID | Value)* Active</feedback>
20 AnswerIdList ::= <AnswerIdList> (INTEGER_LITERAL)+</AnswerIdList>
21 Active ::= <active> (true | false) </active>
```

Figure 2 – The AGLO Meta-model

The question section allows the creation of a question out of static text combined with symbol values. The combined content at design will have the same web representation in the learner's browser.

The answers section is dedicated to expressing the answers of the question. The AGLO model offers short and long text answer on one line or multiple lines and multiple answers for the learner to select. Instead of designing the web representation of the answers we will use formalisms for the designer to express only the answers values and the DSEL platform will decide how they will be presented to the learner. For example, if the designer wants to create 3 answers where 1 correct and 2 incorrect then he will have to specify only this strict information and the framework will generated a form with 3 radio buttons to enable single selection. If we have two correct answers and one false answer then the generated form will contain checkboxes to enable multiple selection. This change of web representations may occur between several instantiations.

The feedbacks section is dedicated for expressing how feedback is created and presented to the student according to the provided answer. Feedback is a feature required actively by students so the efforts to design and generate is motivating and justified. Feedbacks seem to be connected to answers. It is difficult to create dynamic feedbacks because for each answer false or true we need to provide details and to explain the learner what he did wrong or good and why.

The DSEL platform is built of several modules. An authentication module will enable student login using already existing accounts in Facebook, Google and Yahoo portals. The platform contains a user interface module to allow the student to navigate on the competence tree. The tree is dynamically loaded from XML files stored in a directory hierarchy. The platform contains also an AGLO content generator which builds a table of symbols from the AGLO model. The component evaluates the expressions (some based on random numbers) and generates the consumable HTML content. Then the component assesses the correctness of the response and displays the learning experience on the screen.

IV. THE DATA STRUCTURES DISCIPLINE COMPETENCE TREE

The designed competence tree is layered on five levels [28]. On the first level we have the domains that were designed to model entire disciplines. For the data structures and algorithms disciplines we modeled two domains: the former for simple data structures like: arrays, linked lists, hashes and the latter for trees and graphs. Domains comprise further general competences.

General competences are second level competence tree nodes. They refer to the ability to design, implement and use abstract data types (ADT). When such elements tend to be complex we can replicate them on different nodes and attach specific operators. For example, the array ADT will be spread on several nodes dealing with specific operators like: linear search, binary search etc.

Specific competence are third level competence tree nodes. For example in the context of array operators, each learned sorting algorithm will have the following specific competences: to know the algorithm principles: i) to know frequent use cases of the algorithm in practice; ii) to know the algorithm input; iii) to know the algorithm output; iv) to recognize the algorithm in practice; v) to know the algorithm steps; vi) to know the algorithm basic block scheme; vii) to be able to code moves; viii) to understand variable roles; ix) to be aware of different algorithm versions.

Variables are fourth level competence tree nodes. The competence variables will contain several actions having the same learning objective using different methods. All these levels presented so far represent competence refinements and allow the creation of taxonomies.

Actions are on the fifth level of the competence tree and they are they identify with AGLOs. These nodes represent the execution units of the framework and they will be the main learning activity provider.

V. THE xAPI INTEGRATION MODEL

In this chapter we will analyze and design the steps necessary for the integration of xAPI with AGLOs in the context of DSEL platform. To summarize, we need to record experiences from AGLO actions that helps the student achieve a certain competence.

Firstly, we need to identify the learning activity providers. We identify two types of activities: i) student platform interaction activities for orientation purposes like: login, logout, competence tree node accesses, navigation to parent node, to sibling node, jumping to a different node, etc. and ii) learning actions that refer strictly to student - AGLO interactions. Diving into details, we will determine how to generate a statement composed of: actor, verb, object, result, context, authority, timestamp, attachments.

The "actor" field is based on the authentication mechanism implemented in the DSEL platform. The student access in the platform is allowed based on Facebook, Google and Yahoo authentications. The full name, the e-mail address and the provider name will be used for recording the learning experience. Learners may use different accounts for recording learning experiences, so it will be the task of the learning analytics tool to unify the recorded statements. Figure 3 depicts such an example.

```
"actor": {
        "name": "Ciprian-Bogdan Chirila",
        "account": {
            "homePage": "http://www.facebook.com",
            "name": "cipri_chirila@yahoo.com"
        }
    }
```

Figure 3 – Actor Example

The verbs we identified and which are suitable to our needs can have several values like: "logged-in", "logged-out" for the interaction with the platform, "attempted" and "completed" for interacting with internal competence tree nodes, namely, all except action nodes. Figure 4 presents an example in this sense.

```
"verb": {
    "id": "https://w3id.org/xapi/adl/verbs/attempted",
    "display": {
        "en-US": "attempted"
    }
}
```

Figure 4 – Verb Example

Generally, the object of the statement in the context of DSEL will be the competence tree node: domain, general competence, specific competence, variable or action. Non-terminal nodes are navigation or access objects, while terminal nodes are learning and assessment objects.

```
"object": {
       "id": "https://aspc.cs.upt.ro:9443/dsel/dom01-sda",
       "definition": {
              "type": "http://adlnet.gov/expapi/activities/course",
              "name": {
                     "en-US": "Data Structures and Algorithms",
                     "ro-RO": "Structuri de date si algoritmi"
              },
              "description": {
                     "en-US": "Data structures and algorithms lecture",
                     "ro-RO": "Curs de structuri de date si algoritmi"
              },
              "extensions": {
                     "https://aspc.cs.upt.ro:9443/dsel/university":
                     "Politehnica University Timisoara"
              }
       }
}
```


Figure 5 – Object Example

The “result” field has no special customizations. The “completion” and “success” fields have the default semantics, while the “score” field will have scaled values of 0.0 and 1.0 because AGLOs answers can be correct or false. The scaled mark for an AGLO is obtained by computing a mean of the several experiences with the same AGLO.

The “context” statement field contains: i) the competence tree current node and path; ii) the array of symbol values and student answers, expressed as a JSON sub-object. The motivation is be able to reproduce the concrete generated context, to enable offline debugging and to be able to ameliorate the AGLO model.

The “authority” of the DSEL generated statements is based on the authentication provider of the node creator or AGLOs designer, so we can use their names (Facebook, Google, Yahoo) and identifiers for recording.

The “timestamp” is set according to the ISO 8601 standard and in our implementation we will use an “yyyy.mm.dd-hh.mm.ss” format for practical reasons. Therefore, we will be able to use basic string comparisons to compile analytics.

The “stored” field is set at the time of recording the statement in the database table and with our DSEL implementation it should be equal with the previous “timestamp” field.

The “attachments” field is not used in this version of AGLO development.

VI. CONCLUSIONS AND PERSPECTIVES

In this paper we presented the integration steps of a xAPI based LRS into the DSEL platform which manages generative learning objects, namely AGLOs. We mapped the AGLO model on the xAPI vocabulary (Tin Can Registry and ADL Vocabulary) and for a coarse-grained generation of learning experiences we managed to reuse the already existing verbs and activity types.

For a fine grained generation we proposed the solution of adding the domain specific semantics in the object component of the statement.

Overall the vocabularies used cover the basic needs, one observation refers to a slight lack of symmetry. For example, the “exited” verb does not have a clear symmetrical “entered” and instead of it we decided to use “attempted” which we consider having the closest semantics.

As future work we intend to model other refinements and to assess their usefulness in the e-assessment of students learning.

References

[1] Advanced Distributed Learning Initiative, 2017. Experience API, http://adlnet.gov/adl-research/performance-tracking-analysis/experience-api/

[2] Advanced Distributed Learning Initiative, 2017. SCORM, https://www.adlnet.gov/adl-research/scorm/

[3] Bogdan, R., 2016. Guidelines for developing educational environments in the automotive industry, 1st International Conference on Smart Learning Ecosystems and Regional Developments, Timisoara, Romania.

[4] Bogdan, R., Ancușa, V., 2016. Developing e-learning solutions in the automotive industry. World Journal on Educational Technology: Current Issues. 8(2), 139-146.

[5] Boyle, T., 2003. Design principles for authoring dynamic, reusable learning objects. Australian Journal of Education Technology, vol. 19, no. 1, pp. 46-58.

[6] Boyle, T., 2006. The design and development of second generation learning objects. Invited talk at Ed Media 2006, World Conference on Educational Multimedia, Hypermedia and Telecommunications, Orlando, Florida, June 28.

[7] Boyle, T., Bradley, C., 2009. User Guide for the GLO Maker 2 Authoring Tool, http://www.glomaker.org

[8] Burbaite, R., Bespalova, K., Damasevicius, R., Stuikys, V., 2014. Context Aware Generative Learning Objects for Teaching Computer Science, International Journal of Engineering Education, vol. 30, no. 4, pp. 929-936.

[9] Chirila, C.B., 2013. A Dialog Based Game Component for a Competencies Based E-Learning Framework, In proceedings of SACI 2013 8-th IEEE International Symposium on Applied Computational Intelligence and Informatics, pp. 055--060, Timisoara, Romania, May.

[10] Chirila, C.B., 2014. Educational Resources as Web Game Frameworks for Primary and Middle School Students, In proceedings of eLSE 2014 International Scientific Conference eLearning and Software Education, Bucharest, Romania, April.

[11] Chirila, C.B., 2014. Generative Learning Object Assessment Items for a Set of Computer Science Disciplines, In proceedings of SOFA 2014 6-th International Workshop on Soft Computing Applications - Advances in Intelligent and Soft Computing, Springer Verlag, ISSN 1867-5662, Timisoara, Romania, July.

[12] Chirila, C.B., Ciocarlie, H., Stoicu-Tivadar, L., 2015. Generative Learning Objects Instantiated with Random Numbers Based Expressions, BRAIN - Broad Research in Artificial Intelligence and Neuroscience, vol. 6, no. 1-2, Bacau, Romania, October.

[13] Chirila, C.B., Raes, R., Roland, A., 2016. Towards a Generic Gamification of Sorting Algorithms, In Proceedings of 12th International Symposium on Electronics and Telecommunications ISETC 2016, Timisoara, Romania, October.

[14] Damasevicius, R., Stuikys, V., 2008. On the Technological Aspects of Generative Learning Object Development, Third International Conference on Informatics in Secondary Schools - Evolution and Perspectives (ISSEP 2008), pp. 337-348, Torun, Poland, July.

[15] Damasevicius, R., Stuikys, V., 2009. Specification and Generation of Learning Object Sequences for e-Learning Using Sequence Feature Diagrams and Metaprogramming Techniques, In proceedings of 2009 9-th International Conference on Advanced Learning Technologies.

[16] ECMA International, 2016. Standard ECMA-262 ECMAScript 2016 Language Specification, http://www.ecma-international.org/publications/standards/Ecma-262.htm.

[17] Ermalai, I., Drăgulescu, B., Ternauciuc, A., Vasiu, R. 2013. Building a module for inserting microformats into Moodle. Adv. Electr. Comput. Eng. Jan 1;13(3):23-6.

[18] Florea, A., Gellert, A., Florea, D., 2016. Teaching programming by developing games in Alice, The International Scientific Conference eLearning and Software for Education, Bucharest: "Carol I" National Defence University.

[19] Han, P.; Kraemer, B.J., 2009. Generating Interactive Learning Objects from Configurable Samples, emphIn proceedings of International Conference on Mobile, Hybrid and On-line Learning, pp. 1-6, Cancun, Mexico, February.

[20] IEEE Learning Technology Standards Committee, 2016. LOM working draft v4.1 Available: http://ltsc.ieee.org/doc/wg12/LOMv4.1.htm

[21] Jones, R., Boyle, T., 2007. Learning Object Patterns for Programming. Interdisciplinary Journal of Knowledge and Learning Objects, vol. 3.

[22] Jung, H., Park, C., 2012. Authoring Adaptive Hypermedia using Ontologies International Journal of Computers Communications & Control, ISSN 1841-9836, volume 7(2), pp. 285-301, June.

[23] Karpova, M., Shmelev, V., Dukhanov, A., 2016. Information resource based on scientific software as a core of interdisciplinary learning resources, 2016 IEEE Frontiers in Education Conference (FIE), pp. 1-5, Eire, PA, USA.

[24] Naaji, A., Mustea, A., Holotescu, C., Herman, C. 2015. How to Mix the Ingredients for a Blended Course Recipe, Journal of Broad Research In Artificial Intelligence and Neuroscience, 6(1-2), ISSN 2067-3957, Bacau, Romania, October.

[25] Stuikys, V., Burbaite, R., Damasevicius, R., 2013. Teaching of Computer Science Topics Using Meta-Programming-Based GLOs and LEGO Robots, Journal of Informatics in Education, vol. 12, no. 1, pp. 125-142, Institute of Mathematics and Informatics, Vilnius.

[26] Stuikys, V., Damasevicius, R., 2007. Towards Knowledge-Based Generative Learning Objects, Information Technology and Control, vol. 36, no. 2, ISSN 1392-124X.

[27] Stuikys, V., Damasevicius, R., 2008. Development of Generative Learning Objects Using Feature Diagrams and Generative Techniques, Journal of Informatics in Education, vol. 7, no. 2, pp. 277-288, Institute of Mathematics and Informatics, Vilnius.

[28] Vlașin, I., 2013. Competența: participarea de calitate la îndemâna oricui, Editura Unirea, Alba Iulia, Romania.