ANNALS of Faculty Engineering Hunedoara
– International Journal of Engineering
Tome XIII [2015] – Fascicule 4 [November]
ISSN: 1584-2673 [CD-Rom; online]
a free-access multidisciplinary publication of the Faculty of Engineering Hunedoara

[1.] Ciprian-Bogdan CHIRILA

A COMPARISON OF MCQ AND AGLO GENERATIVE LEARNING OBJECT MODELS

[1.] Department of Computer and Information Technology, University Politehnica Timişoara, ROMANIA

Abstract: Generative learning objects are the second generation of learning objects representing instantiable patterns designed for reuse purposes. Among them we identified in the literature two concrete models: Moodle Coordinate Questions and Auto-generative Learning Objects. Each model has its own approach of creation and generation of student consumable learning objects. A comparison and an analysis of the semantic details will help us to improve both models.
Keywords: learning objects, generative learning objects formats, variables, random values

1. INTRODUCTION

Nowadays universities tend to increase the number of students while keeping almost constant the number of teachers. In the area of Information Technology (IT) such cases are quite frequent because of the IT industry high salaries and other benefits. Due to its recent developments the IT domain is more and more attractive for both students and teachers which quit their university jobs and go to work in the IT industry. The solutions for keeping the quality in the process of student training are based on e-learning. In this sense learning objects (LOs) [22] play an important role in modern learning infrastructures based on learning management systems (LMS). Several standards were created for LOs like LOM [14]. One of the most popular LMS is Moodle [16] because it is a free and alive product and because it is very used in academia. Moodle is a free web application as education software that facilitates the creation of modular courses to be delivered online, based on the social constructionist pedagogy.

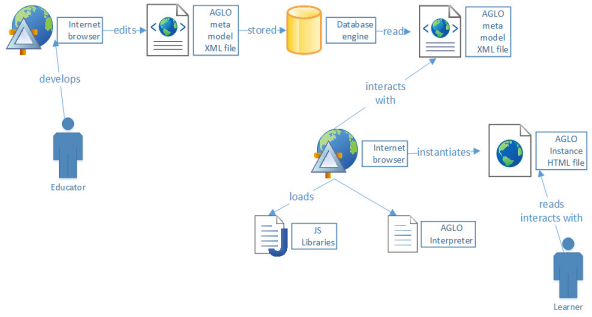


Figure 1 – The GLO Workflow

LOs are digital resources for learning that can be distributed across networks in large or small chunks [22].

GLOs are considered the second generation of learning objects containing instantiable pedagogical patterns targeted for reuse. The GLO main principle belongs to the object-oriented technology being linked to concepts like: class, object and instantiation.

The AGLO approach uses the workflow depicted in Figure 1 [4]. The educator edits the AGLO model file according to a predefined metamodel and semantics. The model is then saved in a database engine in order to be read further by an interpreter through a browser and a web server in order to be consumed by the learner.

In this paper we will discuss the details resulting from the comparison of two GLO models:

i) the first model is from the Moodle Coordinate Question (MCQ) plugin [15] and

ii) the second model is the auto-generative learning object (AGLO) model proposed in [4, 5, 6, 7].

Comparing and analyzing the syntax will help us comparing the semantics and the expressivity of each model. We will also analyze the tooling support for both models. The structures of MCQ and AGLO models are quite similar. They both consist in:

i) variables definition section;

ii) question section;

iii) answer section; and

iv) result or grading section.

They have common structural elements, but the approaches are different maybe because of the different learning objectives: MCQ is dedicated to learn mathematics and physics while AGLOs are dedicated to learn computer science disciplines the core of IT specialist

know-how. The paper is structured as follows. Section 2 describes AGLO structure. In section 3 we compare the details of the MCQ question structure. In section 3 we analyze the facilities for the creation of multiple questions from the very same instance of variables. Section 4 analyzes the way answers are read and assessed from the student. Section 5 shows how variables are defined and how are they instantiated with random values. Section 6 presents related works. Section 7 concludes and sets the perspectives.

2. AUTO-GENERATIVE LEARNING OBJECTS

AGLOs are defined in [Chi2015BRAIN] as both mathematical models and Extended Bachus Naur Forms (EBNF) models. The model was previously refined based on ideas from [5,6,7]. Next, we will present the EBNF model of the AGLO concept defined in [4].

Figure 2 depicts the structure of the AGLO model [4]. The main sections are as shown:

i) name – where the name of the AGLO is given;

ii) scenario – where an informal description of the AGLO is given, and also here are defined and initialized variables based on mathematical expressions involving random numbers. In our model we consider the variables as symbols;

iii) theory – a section where some theoretical examples can be shown, all can be based on the generated variable values;

iv) question – a section where the question is composed out of static text and generated variable values;

v) answers – a section where the answers are defined.

There can be one or multiple answers. The interpreter will decide which is the best presentation strategy, like radio buttons or check boxes, etc; vi) feedbacks – a section where the correct answers are explained and motivated, in their structure we can use again variable generated values. The syntax relies on XML markup language, while the implementation is based on JavaScript [10].

```
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 "</name>"
07 Type ::= "<type>" ("boolean" | "int" | "float" | "double" | "string" |
"array") "</type>"
08 Expression ::= "<expr>" Function "(" ExpressionList ")" "</expr>"
09 ExpressionList ::= Expression (, Expression)*
10 Function ::= (element from functions and operators list of JavaScript)
11 Theory ::= "<theory>" (ID)* "</theory>"
12 Question ::= "<question>" (ID | Value)* "</question>"
13 Value ::= "<value>" "<name>" ID "</name>" "</value>"
14 Answers ::= "<answers>" (Answer)+ "</answers>"
15 Answer ::= "<answer>" "<id>" INTEGER_LITERAL "</id>" (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 – AGLO EBNF Model

3. QUESTIONS

The MCQ editing facilities involve four sections as follows:

i) main question section where the general context of an exercise is described. There are set also the random variables values expressed through discrete enumeration having a step which is implicitly 1 but it can have other values like 0.1 or 2;

ii) sub-question section containing the text, unit and grading criteria. These sub-questions tend to reuse the set of instantiated random variables in the main question, offering the choice to the tutor to develop further its exercise ideas;

iii) extra-options are a set of specific options that apply to all sub-questions;

iv) variables instantiation checking section is a section where a few samples of instantiated questions, optionally with answers are displayed to the tutor in order to have a visual representation of the exercises the student will see.

Currently, the AGLOs are editable in XML format and only at runtime the model is interpreted and instantiated with different random values. The variables instantiation checking section seems to be a valuable idea in the design of generative learning objects. AGLOs are based on random number generators so such a facility would discover eventual errors regarding generated values.

MCQ single questions or sub-questions are formed out of:

i) mark, namely the points earned by the student by solving correctly the exercise;

ii) set of local variables, which are symbols having values and types to be used in forming the question content;

iii) answer type which can be: number, numeric or numerical formula;

iv) grading variables which are variables used in the process of grading the student activity, usually involved in a mathematical formula;

v) grading criteria - modeling the grading formula;

vi) unit is the unit of the answer, usually for mathematics and physics question;

vii) other rules – which will enter further into details;

viii) placeholder name – is a name that can be used to refer the current question in order to relocate it in the context of other sub-questions;

ix) sub-question text – the text of the question or sub-question.

The MCQ multiple question facility is a generalization of the single sub-question facility. In the multiple sub-question configuration placeholders are required in order to locate the sub-question positions. When no placeholders are used then the sub-questions are

stacked in their order of definition. The main question text includes the placeholders where the sub-questions will be placed. The placeholder name must be prefixed by the caret # symbol and enclosed between accolades {} when used in practice.
In our AGLO approach multiple questions can be defined without any special configuration. In both questions and answers sections generated values can be used to be shown. In the answer section input fields are also available for the student to fill in order to assess their content. The MCQ separation of questions seems to be more like a conceptual delimitation of questions based on the very same generated variables.

4. ANSWERS

In MCQ model the answering facility includes multiple answer boxes denoted by fixed names like {_0}, {_1}, etc and {_u} for the unit box. The answer assessment differs with the type of the answer. If the answer is a number then an error variable is computed as a difference between the computed and the given answer. The answer is correct if the difference is smaller than a defined threshold denoted absolute error. For answers of type numeric or numerical formula the mathematical expression is evaluated for all evaluation points of random variables domain. The numeric type is defined as mathematical expression formed out of constants, variables and operators, while the numerical formula type includes also mathematical functions like sin, cos, etc.
In the AGLO model the multiple answers are accepted in the idea of generating different presentation layouts depending on the generated values and the number of correct answers.
For example, if there are multiple answers with multiple correct ones than the presentation form will contain checkboxes so the student can select them. If there are multiple answers with only one correct answer then a form with radio button or a combo box will suffice, thus helping a little bit the student indicating that he has to choose only one. We remind the reader that the content of the answers depend on variables which are generated based on expressions and random numbers and that they are different at every instantiation. Thus, we can introduce variability also at the presentation level, which can be driven by several other factors like difficulty level, student preferences etc.
The other usual case is to have a single answer to be written in a text field or in a text area which is simple to implement. The assessment of the answers in Information Technology disciplines is a simple string comparison with the correct answer of course after trimming the eventual white spaces. This approach doesn't seem to be enough since there are questions with multiple equivalent correct answers that need to be checked by some mathematical formula.

5. VARIABLES AND RANDOM VALUES

In the MCQ model the instantiation process is based on random variables which are meant for all questions. Global variables are created for the main text to be used through substitution. Local variables are designed for sub-questions and answers. Grading variables are created for answer boxes and they are of course used for assessment. The inclusion relation between variables starts with random variables which are included in all variable sets and ends with the grading variables which include all types of variables. This hierarchical approach of the variable system is good for defining related questions. In our approach we have only one level of variables, actually called symbols. Grading variables in our model are expressed as Boolean expressions to be evaluated after the student completed the input boxes.
The idea that each sub-question has its set of variables probably with similar names may create confusion. If the variable names are different then there is no need for individual sets of local variables.
The idea of having a special variable {_u} for the unit is a particular case that breaks the orthogonality of the MCQ model and the author admits that when two input boxes one for numerical answer and the second for unit answer are neighbors then the two boxes are merged into a single one. The problem is what happens if we want to design multiple answers for the very same sub-question having different measurement units.
We consider that in our AGLO model the general approach is orthogonal since any unit answer is treated as an answer in general.
The variables names in both models follow the same general rules that they have to start with a letter or an underscore.
Regarding the types of variables in the MCQ model we have:
i) numbers expressed in several formats, for example in exponential format;
ii) strings enclosed by quotes;
iii) list of numbers associated to arrays;
iv) lists of strings associated to arrays;
v) algebraic variables - as a set of numbers, defined in the non-random variable scope.
The MCQ model types are quite limited compared to AGLO which allows dynamic types created by composing basic ones. For example, we can define an array of structures, facility supported by JSON format and its functions. There is a special facility in the MCQ model regarding integer and float lists, they can be defined by the first element, the last element and optionally an increment to iterate between the two limits.

The MCQ random values are defined by expressions to be computed or by lists to be randomly chosen from. The AGLO approach use only expressions based on a function generating random subunitary values. These basic facilities can be used further to create complex data structures.

In the MCQ model the idea of equal probability for value appearance is developed since in the AGLO model using different formulas we can obtain different probabilities depending on the necessities. For example, in expression random(0,2,0)==0 ? "a" : "b" the probability of getting an "a" is 1/3 while the one of getting a "b" is 2/3.

The syntax for using variables in the text is simpler in the MCQ model based on accolades { }, while in AGLO we use XML syntax, like <value name="alfa">.

6. RELATED WORKS

GLOs are special LOs considered to have a higher degree of reuse. [1, 2, 12] are seminal papers for the GLO concept. The works of Stuikys [18, 19, 20, 21] and Damasevicius [8, 9, 3] present a GLO model based on input knowledge and output knowledge and relying on feature diagrams to assess commonality and variability. [17] shows a depreciation GLO example implemented in the domain of economy, namely in accounting. [11] shows a GLO approach based on Bloom taxonomy cognitive layers with an implementation in XML and Action Script 3.

7. CONCLUSIONS AND FUTURE WORK

In this paper we compared the design and the semantics of two GLO models, namely MCQ and AGLO. The AGLO model tends to be more general that the MCQ model. In AGLO we have defined a set of operators which are freely composable. In the AGLO model we have complex data structures based on primitive types, type projection (array) and Cartesian product (structures). The two level variable system of the MCQ model is not necessary for simple questions.

The MCQ feature of showing samples of instantiated LOs has great potential in showing the tutor the flavor if its exercises. MCQ model definition of sets by ranges is an interesting idea that could be implemented in AGLO using dedicated library functions.

MCQ grading criteria are based on the same principles of writing Boolean expression for student answer assessment.

MCQ statistical error variables to reflect the difference between the answers and the student responses and the unit system tend to be more useful in the area of Mathematics and not so much in the IT domain.

As future work we plan to change the AGLO model and to implement the useful features revealed by this research.

Acknowledgements

This paper is supported by the Sectorial Operational Programme Human Resources Development (SOP HRD), financed from the European Social Fund and by the Romanian Government under the project number POSDRU 159/1.5/S/134378.".

REFERENCES

[1.] Boyle, T. (2003), Design principles for authoring dynamic, reusable learning objects, Australian Journal of Education Technology, volume 19(1), pp. 46-58.

[2.] 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.

[3.] Burbaite, R.; Bespalova, K.; Damasevicius, R.; Stuikys, V. (2014), Context Aware Generative Learning Objects for Teaching Computer Science, International Journal of Engineering Education, volume 30(4), pp. 929-936.

[4.] Chirila, C.B.; Ciocarlie, H.; Stoicu-Tivadar, L. (2015), Generative Learning Objects Instantiated with Random Numbers Based Expressions, Broad Research in Artificial Intelligence and Neuroscience (BRAIN), October, (to appear).

[5.] 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.

[6.] 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.

[7.] 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.

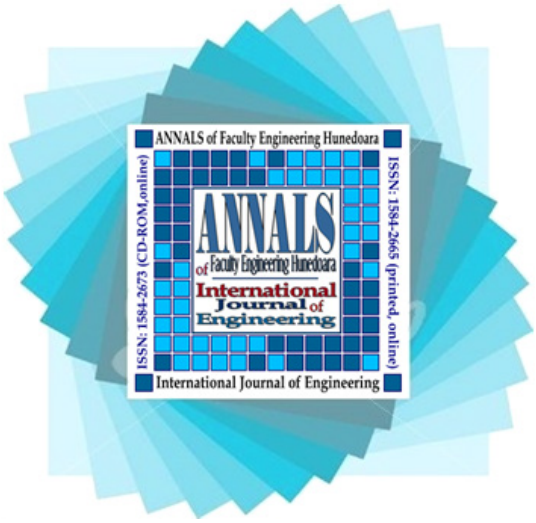
[8.] Damasevicius, R.; Stuikys, V. (2008), On the Technlogical Aspects of Generative Learning Object Development, Third International Conference on Informatics in Secondary Schools - Evolution and Perspectives (ISSEP 2008), pp. 337-348, Torun, Poland, July.

[9.] 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, pp. 572-576, Riga, Latvia.

[10.] ECMA International: Standard ECMA-262 ECMAScript Language Specification Edition 5.1 (2011),http://www.ecma-international.org/publications/standards/Ecma-262.htm.

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

[12.] Jones, R.; Boyle, T. (2007), Learning Object Patterns for Programming, Interdisciplinary Journal of Knowledge and Learning Objects, vol. 3.
[13.] 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.
[14.] IEEE Learning Technology Standards Commitee (2000), LOM working draft v4.1, http://ltsc.ieee.org/doc/wg12/LOMv4.1.htm.
[15.] Moodle Coordinate Question Plugin (2015), Tutorial and Documentation, https://code.google.com/p/moodle-coordinate-question/, accessed in September.
[16.] Moodle Pty Ltd (2015), Moodle Open Source learning platform, http://www.moodle.org, accessed in September.
[17.] Oldfield, J.D. (2008), An implementation of the generative learning object model in accounting, In proceedings of Asciilite (Australasian Society for Computers in Learning in Tertiary Education), Melbourne, Australia.
[18.] Stuikys, V.; Damasevicius, R. (2007), Towards Knowledge-Based Generative Learning Objects, Information Technology and Control, 36(2), ISSN 1392-124X.
[19.] Stuikys, V.; Damasevicius R. (2008), Development of Generative Learning Objects Using Feature Diagrams and Generative Techniques, Journal of Informatics in Education, volume 7(2), pp. 277-288, Institute of Mathematics and Informatics, Vilnius, Lithuania.
[20.] Stuikys, V.; Brauklyte, I. (2009), Aggregating of Learning Object Units Derived from a Generative Learning Object, Journal of Informatics in Education, volume 8(2), pp. 295-314, Institute of Mathematics and Informatics, Vilnius, Lithuania.
[21.] 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, volume 12(1), pp. 125-142, Institute of Mathematics and Informatics, Vilnius, Lithuania.
[22.] Wiley, D. (2000), Connecting learning objects to instructional design theory: A definition, a metaphor, and a taxonomy. D. Wiley (Ed.), The instructional use of learning objects: Online version. Retrieved January 05, 2015 from http://reusability.org/read/chapters/wiley.doc.



ANNALS of Faculty Engineering Hunedoara – International Journal of Engineering