The 14th International Scientific Conference
eLearning and Software for Education
Bucharest, April 19-20, 2018
10.12753/2066-026X-18-000

Auto-Generative Learning Objects for Middle School Arithmetic

Felicia-Mirabela Costea, Ciprian-Bogdan Chirila
Department of Computers and Information Technology, Politehnica University, Piaţa Victoriei Nr. 2, Timişoara, România
mirabela.costea@cs.upt.ro, chirila@cs.upt.ro

Vladimir-IoanCretu
Department of Computers and Information Technology, Politehnica University, Piaţa Victoriei Nr. 2, Timişoara, România
vcretu@cs.upt.ro

***Abstract**: Nowadays middle school students tend to use more and more gadgets like tablets, smart phones and laptops. Learning materials tend to become electronic having automatic evaluation mechanisms like quizzes, tests, etc. The learning materials usually have static content. Even LMS supported evaluations implemented by quizzes have static questions and answers and the only solution to create diversity for the student is to shuffle the order or both questions and answers. In this context there is a high interest in the topic of reusable learning designs in order to create dynamic content. These offer the prospect of capturing effective designs for learning and making them available for reuse and adaptation. The idea of capturing successful learning designs and making these the basis for reuse, rather than content, is at the core of the concept of generative learning objects or GLOs. AGLOs are automatically instantiate GLOs based on a meta-model expressed in XML, domain JavaScript object libraries and a generation environment implemented in JavaScript. The reuse of AGLO requires limited or no programming skills at all. In this paper we propose a set of AGLOs dealing with middle school arithmetic: transformations, additions, subtractions, multiplications, divisions of rational numbers as fractions. The tutors can change the presentation, can adapt the existing design to their own needs, or can write new functions and objects to extend the domain library. The prototype is implemented as a JavaScript application running on Apache web server connected to a MySQL database. The approach was tested online with the fifth-grade middle school students.*

***Keywords:** Auto-generative learning objects; arithmetic; fraction operations; middle school student.*

I. INTRODUCTION

If we look at young people in modern society, we find that they are almost technology dependent. They use electronic devices for learning, socializing, in leisure time. It is important for the education to keep up with this trend.

Learning is a process and it is part of our daily lives. Modern technology has made it simple for students to learn from anywhere through online education and mobile education. Also, students now use modern technology in classrooms to learn better. This has made learning more convenient and fun. Also new modern educational technologies support individual learning which gives a chance to students to learn on their own with no need for tutors.

There is considerable interest in the topic of reusable learning designs. These offer the prospect of creating effective designs for learning and making them available for reuse and adaptation. Much of this work is focused at the level of activities for studying key concepts for middle school arithmetic.

The idea of capturing successful learning designs and making these the basis for reuse, rather than content, is at the core of the concept of generative learning objects (GLOs) [6],[22],[7]. The traditional approach for developing learning objects has focused on content, and standards for

packaging and describing this content [21], [2], [1]. Repositories of learning objects based on these standards are meant to improve learning: "… by making content more readily available, by reducing the cost and effort of producing quality content, and by allowing content to be more easily shared" [17].

This vision for learning objects has been to improve the quality of teaching and learning through the widespread availability of "self-contained" learning resources. Several repositories have been developed; however, the evidence points to limited impact in achieving this aim [24]. There are several reasons why the learning object vision has failed so far to achieve its potential. In the traditional, standards-oriented approach, there is really no guidance how to develop high-quality learning objects, either in terms of pedagogy or the design features that will facilitate reuse.

In order to solve the shortcomings encountered by the learning objects, we had used auto generative learning objects (AGLOs) which are based on numbers generated randomly, every time it is used, a new exercise is actually generated. Learning objects are instantiated automatically, and the answer and the feedback are also automatic.

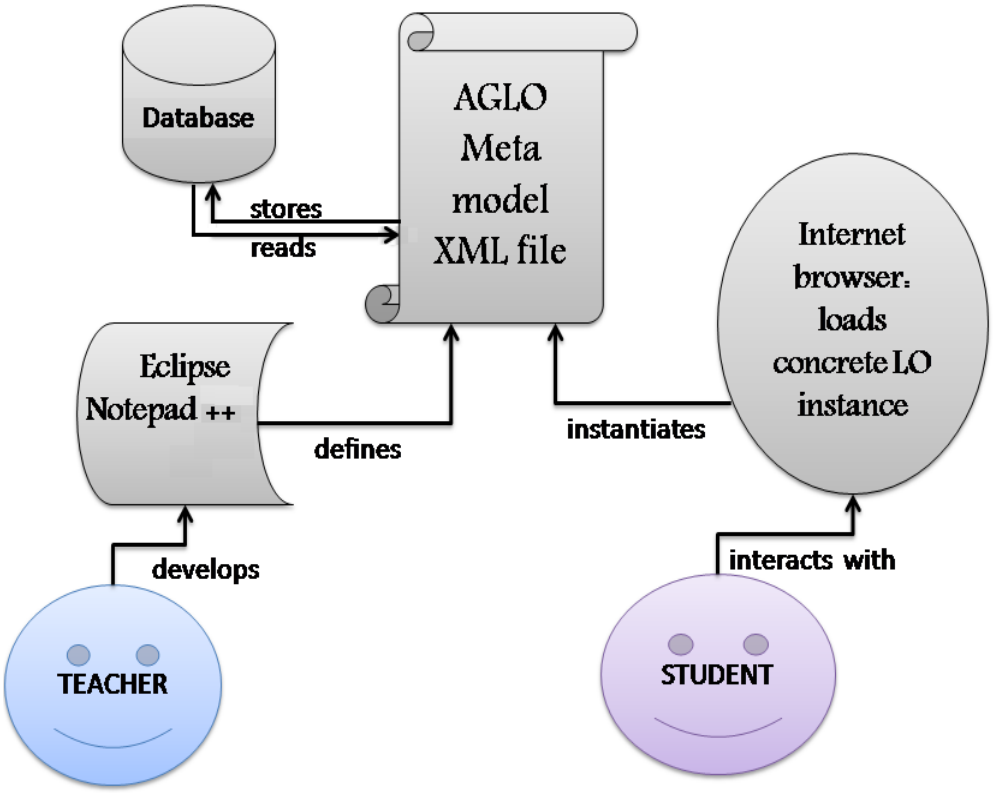

Figure 1. The AGLO

Figure 1 illustrates how our learning object works. The educator develops the meta-model of the learning object with the help of accessible environments: Eclipse and Notepad ++. Editing consists of writing formalisms and using specific editing tools. The model is saved in a MySQL database from where is instantiated. The learner has access to the learning object through a web browser connecting to a web server. When accessing it through the browser, an instance of the learning object is created in HTML having an interactive form.

The paper is structured as follows. Chapter II presents related works in the field of learning objects and generative learning objects. Chapter III presents our model of the AGLO. Chapter IV presents the AGLO model applied in the field of working with fractions at the 5th grade level. Chapter V presents a case study and the obtained results. Chapter VI concludes and sets up the perspectives.

II. RELATED WORKS

The concept of learning object [26] refers to reusable, transferable digital materials to be used for learning purposes. Standards were created in this sense e.g. Learning Object Model (LOM) [21], [2].

GLOs are considered to be the second generation of LOs in this sense. The term of GLO was given by Tom Boyle [6], [7], [8]. They implemented the idea of developing consumer LOs following a reusable pattern and filling in with content.

Some examples of GLOs are given in [19] where is implemented the Bloom taxonomy with pedagogical parameters or in [25] where is achieved a depreciation model from economy.

With our AGLOs we keep the pattern philosophy of GLOs but the generation is different [13]. Our AGLO approach use metaprogramming for content generation but the main difference is that our generation is highly based on random numbers. Applications of the AGLO concept for primary and middle school students are present in [11], [12].

[13], [14] present applications of AGLOs in disciplines from academia, like data structures and algorithms.

[9], [16] present applications of GLOs in Computer Science generated with the help of sequence feature diagrams and meta-programing.

[23] presents a different kind of GLO that changes the content structure based on parameters but also the way of learning and evaluation.

[3], [5], [4] present specific e-learning solutions for training automotive company employees.

Pedagogically, there is also value to using electronic devices in class. One study [10] indicate improvements to active learning and student engagement with content through the use of applications which requires personal devices for students, especially when that use was continued outside the classroom.

III. AUTO-GENERATIVE LEARNING OBJECTS IN A NUTSHELL

The AGLO model was developed in [13]. The purpose of this model was to allow the expression of variable questions, responses and feedbacks for practical problems built up as dialogue games for students in primary and secondary education. The main idea is to use instantiated expression variables based on random numbers. Thus, we can generate values to suit a particular scenario, where the student can practice its skills.

In order to test the model, for example, understanding the operations with simple fractions, we randomly generated two simple fractions that the student has to add, subtract, multiply or divide according to the chosen exercise. If it fails and resumes the exercise, it generates another instance of the variables, thus creating another exercise. As a result of how many times the exercise resumes, another set of variables is instantiated, which helps students to improve their abilities.

Technically speaking, variables, called symbols, are instantiated with JavaScript compatible expressions [18], then these values are joined with inline strings to form AGLO sentences. The student's response is instantiated and stored in variables to be verified using the correct calculations in order to provide student feedback and notes.

Structurally AGLO model has six sections:

i) name – the AGLOs name; ii) scenario – the section where the variables are defined and initialized based on random numbers; iii) theory – a brief theoretical presentation of what should be applied in the exercise; iv) question – the section where a question is made from static text and values of randomly instantiated variable; v) answer – the section where the answer is read from the student and is assessed automatically; vi) feedback – the section where a feedback text is composed out of static text which has the value to explain the student the essence of the exercise.

The model is defined using the EBNF meta-language resulting in the concise grammar from Figure 2.

In the scenario (lines 03-07) the educator writes the main ideas of the learning object in natural language. By ID we denote the identifier token and by CT we denote any constant or literal.

Here too we define the symbols that give a particularity from a dynamic point of view. These symbols will be the parameters of the following section. A symbol has several properties: name, type

and is initialized based on random numbers. The name is an identifier, the type can be: “integer”, “string” or “fraction”.

Expressions are made up of operators and functions (lines 08-10). To be as accessible and portable as possible is based on JavaScript expressions that are evaluated with JavaScript functions.

The theory section (lines 11) is only formed from static data. Here is the concept that applies to solving the problem outlined in the next section.

The question (lines 12-13) contains both static data and dynamic values. Even if the question refers to the same theory, for every instance there is a different exercise. This section creates a dynamic content which is presented to the student.

```
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>” (“integer”|”string”|”fraction”) “</type>”
08 Expression ::= “<expr>” Function “(“ExpressionList”)” “</expr>”
09 Function ::= (element from functions and operators of JavaScript using
random numbers)
10 ExpressionList ::= Expression (, Expression)*
11 Theory ::= “<theory>” (ID)* “</theory>”
12 Question ::= “<question>” (ID| Value)* “</question>”
13 Value ::= “<value>” “<name>” (ID)* “</name>” “</value>”
14 Answer ::= “<answer>” (Answer)+ “</answer>”
15  Answer  ::=  “<answer>”  “<id>”  INTEGER_LITERAL  “</id>”  (ID|Value)*
Correctness “</answer>”
16 Correctness  ::= “<correct>” (“true” |”false”) “</correct>”
17 Feedbacks ::= “<feedbacks>” (Feedback) “</feedbacks>”
18 Feedback ::= “<feedback>” (ID)* “</feedback>”
```

Figure 2. Auto-generative Learning Object Model Definition

The answer (lines 14-16) also contains both static data and dynamic values.

The feedbacks section (lines 17-18) is only formed from static data. The feedback has the role of facilitating learning.

IV. THE AGLO MODELS FOR FRACTION OPERATIONS

The fractions used in exercise are generated randomly. For both simple fractions and decimal fractions, we have created a random instance so that whenever an exercise is accessed, the student works with new numbers.

To begin with, we created a few exercises that looked at the classification, comparison, amplification and simplification of the decimal fractions. The first XML element is reserved for the name of AGLO. The second element is the scenario containing the AGLO description and a set of symbols. In the first AGLO exercise the description from the scenario section is: “A simple fraction is generated. Compare the counter and denominator to the following variants:

- if the count is lower than the denominator, then is a proper fraction;
- if the count is bigger than the denominator, then is an improper fraction;
- if the counter and the denominator are equal, then is a unit fraction. “

```
<scenario>
      ...
      <symbol name="nr" type="integer">random(1,20,0);</symbol>
      <symbol name="num" type="integer">random(1,20,0);</symbol>
      <symbol name="f" type="fraction"> randomFraction2(v("nr"),v("num"));
      </symbol>
      <symbol name="answer" type="string">v("f").classify()</symbol>
</scenario>
```

Figure 3. AGLO Scenario symbols for classification

The description, as seen above, is expressed in natural language and contains the solving steps of the item. The symbols (see Figure 3) will be the parameters of the following section, and their value will be particular to each instance.

In the theory section (see Figure 4) there are presented the theory that is applied in the exercise. The question section is composed out of both static text and elements defined above. They are replaced with the appropriate value so that the learning content to be dynamic.

```
<theory>
	<text>
		For the m / n fraction we have: if m>n is an improper fraction;
		if m=n is an unit fraction; or if m<n is a proper fraction.
	</text>
</theory>
<question>
	<text>
	  Determine what kind of fraction is <value name="nr"/>/<value name="num"/>.
	</text>
</question>
```

Figure 4. AGLO Theory and question for classification

The <answers> XML element (see Figure 5) is composed out of a single answer. This element can be composed out of both text and dynamic values depending on the exercise. The feedback section is static text only and has the role of supporting effective learning.

```
<answers>
	<answer id="1">
		<text>
	<value name="answer"/>
		</text>
	</answer>
</answers>
```

Figure 5. AGLO Answers for classification

The second exercise compares two simple fractions. Two simple fractions are generated for comparison, and the answer is the lower sign, the bigger sign or equal.

For amplification we generated a simple fraction and a number, the number with which we amplify. For simplification we generated only the simple fraction, the number with which it will be simplified is calculated by the “divisor” function, that number is the greatest common divisor of dominator and count.

In exercises 5, 6, 7 and 8 we implemented the operations with simple fractions: adding, subtracting, multiplying and dividing. For any operation there are generated two simple fractions and are calculated according to the formula the corresponding result. The answer is the simplified fraction obtained from the result that is was calculated.

For example, in the addition AGLO the description includes the text: “Two simple fractions are randomly generated, for example 2/3 and 1/4. Calculate the count and the denominator, (2*4+1*3)/(3*4). It simplifies the result by obtaining an irreducible fraction.” The scenario symbols for the gathering of ordinary factions are represented in Figure 6.

In the ninth item we put into practice raising power of fractions. It is thus randomly generated a simple fraction and one number that will represent the exponent. The answer is the result of the rise to power.

For each of the exercises from 1 to 9, 14 and 15, the constructor function randomFraction2() is called, which builds simple fractions.

Fourteenth exercise calculates an improper fraction from a whole plus a part, that is, a whole and a simple fraction are random generated. The 15 exercise calculate the opposite, a mixed numeral from an improper fraction.

Exercise 10 calculates a percentage of a given whole number. Both the percentage and the number are randomly generated. The answer is a real number with two exact decimals.

```
<scenario>
       ...
       <symbol name="nr1" type="integer">random(1,20,0);</symbol>
       <symbol name="num1" type="integer">random(1,20,0);</symbol>
       <symbol name="nr2" type="integer">random(1,20,0);</symbol>
       <symbol name="num2" type="integer">random(1,20,0);</symbol>
       <symbol name="f1" type="fraction">
              randomFraction2(v("nr1"),v("num1"));</symbol>
       <symbol name="f2" type="fraction">
              randomFraction2(v("nr2"),v("num2"));</symbol>
       <symbol name="count_ad" type="integer">
              v("f1").denominator*v("f2").count +
              v("f2").denominator*v("f1").count;
       </symbol>
       <symbol name=" denominator_ad" type="integer"> v("f1").denominator*v("f2").
              denominator;</symbol>
       <symbol name="f" type="fraction">
              randomFraction2(v("count_ad"),v("denominator_ad"));
       </symbol>
       <symbol name="answer" type="string">
              v("f"). count /v("f").divisor()+"/"+
              v("f"). denominator /v("f").divisor(); </symbol>
</scenario>
```

Figure 6. AGLO scenario for addition

We also used the transformation of decimal fractions into simple fractions. For decimal fractions we created a constructor function randomFraction3() (see Figure 7), which is used in exercises 11, 12 and 13. In those ones we have implemented the conversion of a finite decimal fraction, a simple periodic decimal fraction, and a mixed periodic decimal fraction into a simple fraction.

```
<scenario>
       . . .
       <symbol name="pi" type="integer">random(10,20,0);</symbol>
       <symbol name="pzn" type="integer">random(10,20,0);</symbol>
       <symbol name="pzp" type="integer">random(10,20,0);</symbol>
       <symbol name="f" type="fraction">
              randomFraction3(v("pi"),v("pzn"),v("pzp"));</symbol>
       <symbol name="denominator" type="integer"> v("f").denominator;</symbol>
       <symbol name="count" type="integer">v("f").count;</symbol>
</scenario>
```

Figure 7. AGLO scenario for the twelfth item

If the fraction is finite then the periodic part is void, and if the fraction is a simple periodic one then the irregularity decimal part is void. The answer for each item is a simple fraction.

V. CASE STUDY AND EXPERIMENTAL RESULTS

For the content specific functionality, we designed a module named fractii.js which contains the constructor functions and all the other functions that are needed in the fraction domain that we had studied, all listed in Table 1.

Function randomFraction2() is a constructor that creates based on two random numbers a simple fraction. Function randomFraction3() is also a constructor but this return a decimal fraction. Starting from a decimal fraction is calculated a simple fraction with the function calculatesDominatorAndCount(). The function comparison() is used to compare two simple fractions. The function divisor() returns the greatest common factor of dominator and count. AtPower() is a function that calculates the power of a simple fraction. The function classify() makes a classification of fractions by comparing the dominator and the count. The integers() is a function used to get the whole of an improper fraction

As we can see ins the table that there are functions that are used in many exercises having a high reuse rate of 73%, but also there are functions that are used in only one exercise having low reuse rates.

Function:	*Random Fraction2*	*Random Fraction3*	*Calculates Denominator AndCount*	*comparison*	*divisor*	*atPower*	*classify*	*integers*
AGLO 1	✓						✓	
AGLO 2	✓							
AGLO 3	✓				✓			
AGLO 4	✓			✓				
AGLO 5	✓				✓			
AGLO 6	✓				✓			
AGLO 7	✓				✓			
AGLO 8	✓				✓			
AGLO 9	✓					✓		
AGLO 10								
AGLO 11		✓	✓					
AGLO 12		✓	✓					
AGLO 13		✓	✓					
AGLO 14	✓							
AGLO 15	✓							✓
Total uses:	11	3	3	1	5	1	1	1

Table 1. The degree of reuse of the library functions

To evaluate the knowledge of the fifth-grade students at the end of the Fraction chapter we used AGLOs. We then applied an impression questionnaire for the analysis calculations and to draw the conclusions.

Most students said they use tablet or laptop in learning activities 72.72%, while 18.18% do not use them. A significant percentage of 63.63% of the respondents said that this approach is very interesting and that they would like teachers to integrate them into their e-learning materials, and 9.09% of them considered it not useful in their learning process.

VI. CONCLUSIONS AND PERSPECTIVES

The paper presents a learning object AGLO useful for the students in the fifth and sixth grades. The model is useful in working with functions. Students benefit from the interactivity and variability of it, by random instantiation. Educators therefore have new tools available to develop dynamic learning objects at different levels of difficulty. The model is portable and can be used on multiple platforms. Given the results we can estimate that the majority of students will appreciate this kind of exercises.

As a future work we intend to integrate the AGLO concept into other levels and chapters in mathematics as well as in computer science. Another direction is research into the development of facilities offered by such learning objects.

References

[1] Advanced Distributed Learning Initiative, 2018. *Experience API*, http://adlnet.gov/adl-research/performance-tracking-analysis/experience-api.

[2] Advanced Distributed Learning Initiative, 2018. *SCORM*, https://www.adlnet.gov/adl-research/scorm.

[3] Bogdan, R. 2017. *Sentiment Analysis on Embedded Systems Blended Courses*. Brain-Broad Research In Artificial Intelligence and Neuroscience, Volume 8, Issue 1, pp. 35-41.

[4] Bogdan, R., 2016. *Guidelines for developing educational environments in the automotive industry*, Interaction Design and Architectures, Issue: 31, pp. 59-73.

[5] Bogdan, R., Ancuşa, V., 2016. *Developing e-learning solutions in the automotive industry*. World Journal on Educational Technology: Current Issues. 8(2), 139-146.
[6] Boyle, T., 2003. *Design principles for authoring dynamic, reusable learning objects*. Australian Journal of Education Technology, vol. 19, no. 1, pp. 46-58.
[7] Boyle, T., 2009. *Generative learning objects (GLOs): design as the basis for reuse and repurposing*, First international conference of e-learning and distance learning, Riyadh
[8] Boyle, T., Bradley, C., 2009. *User Guide for the GLO Maker 2 Authoring Tool*, http://www.glomaker.org
[9] 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.
[10] Chawinga, W., 2017. *Taking social media to a university classroom: teaching and learning using Twitter and blogs*, International Journal of Educational Technology in Higher Education 14.1: 1-19.
[11] 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.
[12] 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.
[13] Chirila, C.B., 2015. *Auto-Generative Learning Objects for IT Disciplines*, In Proceedings of the International Conference on Virtual Learning 2015, ISSN 1844-8933, Timisoara, Romania, October.
[14] Chirila, C.B., 2017. *Auto-generative learning objects in online assessment of data structures disciplines*, BRAIN - Broad Research in Artificial Intelligence and Neuroscience, vol. 8, no. 1, Bacau, Romania, April.
[15] 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.
[16] 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.
[17] Duval, E., Hodgins, W., Rehak, D. & Robson, R., 2004. *Learning objects symposium special issue guest editorial*, Journal of Educational Multimedia and Hypermedia. 13, 4, 331-342, AACE.
[18] ECMA International, 2017. *Standard ECMA-262 ECMAScript 2016 Language Specification*, http://www.ecma-international.org/publications/standards/Ecma-262.htm.
[19] Han, P.; Kraemer, B.J., 2009. *Generating Interactive Learning Objects from Configurable Samples*, emph In 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] IMS (2009) IMS Global Learning Consortium Web site: http://www.imsglobal.org.
[22] Jones, R., Boyle, T., 2007. *Learning Object Patterns for Programming*. Interdisciplinary Journal of Knowledge and Learning Objects, vol. 3.
[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] Koppi, T., Bogle, L. & Lavitt, N., 2004. *Institutional Use of Learning Objects: Lessons Learned and Future Directions*, Journal of Educational Multimedia and Hypermedia, 13, 4, 449-463, AACE.
[25] Oldfield , J. D., 2008. *An implementation of the generative learning object model in accounting*. In proceedings of Asciilite, Melbourne.
[26] 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.