HOW TO INTEGRATE GENERATIVE LEARNING OBJECTS INTO TEACHING AND LEARNING PROCESSES

Vytautas Stuikys¹, Ilona Brauklyte², Robertas Damasevicius¹

¹ Kaunas University of Technology, Software Engineering Department, Studentu-50, Kaunas, Lithuania, vytautas.stuikys@ktu.lt, damarobe@soffen.ktu.lt
² Klaipeda University, Informatics Engineering Department, Bijunu-17, Klaipeda, Lithuania, ilona@ik.ku.lt

Abstract. We discuss a problem of integrating learning object (LO) instances, which are generated from a generative learning object (GLO), into teaching and learning processes. GLO is such a specification that implements variable and common features of similar (related) LOs, thus enabling to generate instances on demand automatically, if the technology used for the specification allows that. We describe a general framework for creating of GLOs and integrating their instances into teaching and learning. The integrating mechanism we propose uses: 1) a sequence of processes (e.g., generation, selection and ordering of instances) to create sessions for the teaching/learning, and 2) models specifically developed in this context. Our contribution is 1) an integrating mechanism and its models, and 2) the enhanced capabilities to implement the ‘learning by doing’ paradigm using the generative technology and GLOs. The approach is supported by examples and is motivated by a case study that includes GLOs from Computer Science subjects. GLOs are specified using meta-programming as a generative technology.

Keywords: learning object (LO), generative learning object (GLO), learning by doing, creating of LOs on demand, integrating GLOs.

1 Introduction

The concept of Learning Object (further LO or LOs) is a central topic in e-learning research because all activities and processes, implicitly or explicitly, relate to this concept [12]. LOs are defined as small, stand-alone, mediated content “chunks” that can be reused in multiple instructional contexts, serving as building blocks to develop higher-level compounds (e.g., lessons, modules, etc.) [13, 17, 18]. When reused, such units are combined in various ways leading to the great variability of learning content. Variability among a given class of LOs can be conceived as a property when a member of the class has the same or similar attributes, but their values are different. Based on this property of similarity, one instance can be derived from another using some well-established mechanism. Conceptually, similar instances can also be combined together in a unique specification using some well-established integration technology. This idea has lead to the introduction of the new kind of LOs recently, which due to the works of T. Boyle, D. Leeder, R. Morales and their colleagues, are known in the literature [1, 11] as generative learning objects (GLOs).

From the structural viewpoint, GLO is a compound that incorporates a set or family of the related LO instances. From the generative technology viewpoint, GLO is a specification that expresses or integrates variable and common features of similar LOs, thus enabling to generate instances on demand automatically, if the technology used for the specification allows that. The main reason of introduction of GLOs is to achieve a higher level of reuse through generative reuse, which gives higher productivity and better quality.

As the field of GLOs is young enough there are many aspects to be yet researched and investigated. One topic we discuss in this paper is of how GLOs should or might be integrated into teaching and learning processes. The paper’s contribution is 1) mechanisms to integrate LOs into teaching and learning, which are implemented through the use of sequence of processes (e.g., generation of instances, their selection, ordering) supported by the introduced adequate models for teaching and learning; 2) showing capabilities of generative technology to incorporate into the GLO’s structure such variability attributes that support the ‘learning by doing’ paradigm at a much larger extent in comparison with the traditional approaches.

The structure of the paper is as follows. Section 2 analyses the related works. Section 3 provides the definition of GLO and outlines its basic properties. Section 4 deals with variability aspects in e-learning with the emphasis on the variability dimension to support ‘learning by doing’. Section 5 presents a general framework related to the development and integration of GLOs into teaching and learning; the framework focuses on the relevant processes and models. Section 6 presents a case study to approve the proposed ideas. Section 7 summarises and evaluates the results. And finally, Section 8 formulates conclusions and future work.
2 Related Works

We categorize the related works, which are most relevant to our paper, into Stream 1 and 2 as follows.

Stream 1. Reusability of LOs is a fundamental concept in e-learning, thus there are an extremely large amount of sources. We restrict ourselves by presenting only books and reviews and also selected papers, from which the reader can extract the relevant knowledge on LOs reuse. For example, the book [9], which is a collection of papers dealing with various methodological aspects on e-learning, also takes the reader through creating, designing effective LOs, transforming existing content into reusable LOs. The book [12], for example, analyses the repository aspects, as well as describes design of LOs and psychological principles of reusable LOs. The review paper [14] analyses various reuse aspects, including compositionality, granularity and scope of reuse. In our view, those characteristics reflect and represent reusability of LOs at the largest degree. The paper [7] connects LOs with different pedagogical paradigms.

Stream 2. As it was already stated in the introduction, T. Boyle, D. Leeder, R. Morales and their colleagues are innovators who have introduced the concept of GLOs, and discussed its role in e-learning [1, 2, 11]. A conceptual framework letting to understand and adapt GLOs is discussed in [2]. Generative technology (GT) is a software-oriented technology that enables to implement generative components and program generators. As GLO is a domain-specific generative component, GT fits to implement GLOs, too. There are the following kinds of GT: template-based technology [15], generative programming [4], aspect-oriented programming [8], and meta-programming [16]. Selection of the technology to design GLOs usually depends on the designer’s flavor or on his/her experience of using the higher-level programming techniques.

3 What is Generative Learning Object (GLO)?

It is not an easy task to give a short and precise definition of GLO, because it elaborates multiple aspects, such as technological, methodological, pedagogical and e-learning. The definition difficulties are also due to the fact that the term LO (it, of course, tightly relates to GLO) is not defined uniformly in the literature. We accept the definition of the term LO which was taken from [13, 17, 18] and was given in the introduction. That is why we suggest to use a multidimensional scheme for the definition of GLO here.

- From the technological perspective, GLO is a higher-level program, i.e. an executable specification developed using some generative technology. In that aspect, GLO can be also conceived as a program generator allowing generating LO instances on demand (the meta-designer’s view). Various technologies can be used for implementing GLO (e.g., template, generative programming, aspect-oriented programming, meta-programming).
- From the methodological viewpoint, GLO is a highly reusable structure that enables ensuring higher productivity and quality of LOs, thus focusing not only on component-based reuse but also moving methodological efforts in the e-learning domain towards generative reuse, which focuses on variability of the domain.
- From the instructor’s (teacher’s) and learner’s viewpoint (pedagogical perspective), GLO is a set of related LO instances (or a qualified sequence of LO instances).
- From the e-learning perspective, GLO is an extension of the LO concept, which is the fundamental concept of e-learning, in the technological, methodological and pedagogical aspects, thus bringing new capabilities for e-learning and also rising new challenges that should be understood and studied.

As GLO is an executable specification it has a well-established structure (see example in Figure 1).

<table>
<thead>
<tr>
<th>Legend: data between &quot;...&quot; – for user only; language - meta-data or parameter with its value</th>
<th>meta_case – a meta-language function for generalization of GLO</th>
</tr>
</thead>
<tbody>
<tr>
<td>Meta-data as interface</td>
<td>&quot;Aim: to teach by example differences of assignment statements of programming languages&quot;</td>
</tr>
<tr>
<td>Meta-body as a generalized content in variability aspects</td>
<td>&quot;Select language&quot; {1-C++, 2-Pascal, 3-VHDL}</td>
</tr>
<tr>
<td>language:=2;</td>
<td>language={ y=a+b; }, { y:=a+b; }, { y&lt;=a+b; }</td>
</tr>
<tr>
<td>a)</td>
<td>meta_case</td>
</tr>
<tr>
<td>c) y:=a+b;</td>
<td>b)</td>
</tr>
</tbody>
</table>

Figure 1. Model (a), example of GLO (b) and LO instance (c) that corresponds to value 2.
A structural model consists of two basic units: meta-data, also called meta-interface, and generalized structure or meta-body of GLO that implements the anticipated requirements. Meta-data serves for managing the generation process. Meta-body implements variability combined with domain functionality, comprising various attributes (e.g., those discussed above), relationships and constraints. In the context of GLOs, the term variability is essential as it follows from Section 4.

4 What is variability in e-learning domain and how it relates to GLO?

Variability is a technology-based term which is widely exploited in software engineering or Computer Science (for details see, e.g., [3]) meaning variants of features of domain artefacts to be implemented into a program. In the context of e-learning, variability means variants of content attributes, pedagogical attributes, social attributes, etc. In the context of GLOs, variability means the implemented variants of the above stated attributes that were anticipated and included into the requirements of a GLO at its development phase. A variant should be understood as a characteristic or a value of the given attribute. A key property of variability is its ability to be measured. In other words, variability has quantitative characteristics. Even if attributes are pure non-technical (e.g., pedagogical, social, etc.), we can, in somewhat way, express them through some measurable characteristics. For example, we express quality of knowledge through assessment marks (excellent, very good, good, adequate, etc.). We express abilities of students in a similar way (very good, good, average, and poor).

How can we express variability of a teaching content quantitatively? From this perspective, the capabilities are extremely wide: granularity level (size) of knowledge chunks (units); a number of units within a LO; type of units in terms of representation (text, picture, graphic, slides, etc. and various combinations of that); type of units in terms of internal characteristics of the learning content that depends upon the topic (e.g., efficiency, time, strength, weight, etc.); various relationships among characteristics including constraints.

How can we express variability of a learning process quantitatively? Here we restrict ourselves by analyzing the paradigm ‘learning by doing’ only and also we deliver it in a simplified manner. As an introduction to this, one should know the following statement: ‘learning by doing’ has the highest assessment rank in terms of capabilities to remember and thus to gain knowledge. Also one should take into account that there are different figures of how people remember facts. For example, according to the Northwestern University (Chicago) findings, people remember 20% of what they are seeing, 40% of what they are seeing and hearing and 70% of what they are seeing, hearing and doing. According to E. Dale’s Cone of Experience [10], people generally remember: 10% of what they read, 20% of what they hear, 30% of what they see, 50% of what they hear and see, 70% of what they say and write and 90% of what they do.

Figure 2 (adopted from the T.H. Davenport’s book [6]) presents evaluations of the problem only qualitatively. The evaluations are expressed by levels.

![Figure 2. Skills levels in teaching and learning](image)

In this context, GLO can be created with a special focus on variability to support ‘learning by doing’ as it will be demonstrated in our case study. The process of ‘doing’, for example, can be measured through levels such as: mandatory trial level, optional trial level, experimental level (see illustration of our case study, Figure 7), or even research level. Levels differ in complexity or form allowing making a choice by the user. The research level is the most complex activity, which anticipates performing a real research experiment. This level can be reformulated as learning by designing or creating. From the perspective of knowledge level gained by a learner, it is the highest level of knowledge that the learner can achieve.

We summarize the above presented discussion from the perspective of cognitive science. As the variability model describes and represents various attributes of features, their relationships and constraints (what can be summarized as a knowledge model), the variability model can be also treated as ontology [5]. More
5 A framework for Integrating GLOs into Teaching and Learning Processes

5.1 A general description of the framework

The framework includes two phases: development phase and use phase (Figure 3). The first phase outlines the development aspects without specific details because its aim is to explain the context for the second phase. The second phase describes managing tasks for integrating GLOs into teaching and learning process in more details.

The development aspects of GLOs include analysis, specification, design and implementation. At the analysis stage the scope of analysis should be identified first assuming that the topic (theme) for learning is already given. Then analysis focuses on various aspects of variability (e.g., content variability, pedagogical variability and social variability). The results of analysis are artefacts for the specification of requirements and construction of the variability model.

At the specification stage, the following activities are performed: requirements statement and analysis, a more precise description of variability attributes, meaning the classification and identification of attribute values (characteristics), identification of relationships and constraints between variability attributes. The results of specification are used for creating a variability model for the next stage. Various forms for representing the model can be used (e.g., narrative description, UML, feature diagrams, etc.).

At the design stage, the following tasks are considered: the development or choice of a structure of GLO, selection of the implementation technology and analysis of its capabilities, and creation of rules for transforming the specification into implementation.

At the implementation stage, the following tasks are considered: combining the transformation rules with the technology, coding work, testing and, perhaps, experimental work. The result is a complete documentation of a GLO, which is the executable specification supported by the environment of the technology.
5.2 A description of integrative aspects

The integrative aspects are outlined in the given framework as the use phase (see Figure 3). It is important to note that the use phase may be connected directly with the development phase (e.g., if a designer and a learner (teacher) are the same person, say, a researcher, who performs experiments). But the typical case is that in which the development phase and the use phase of GLOs are separated and the actors are different persons. Let us consider this typical case. A GLO is extracted from the external repository by an instructor or a learner. The next action what he/she need to do is the initiation of the generation process to produce LO instances. A GLO usually contains much more functionality (e.g., a larger extent of variable features that are needed in the concrete context of use). GLO of a given topic, which was created by the designer after the development phase, and GLO for the same topic at the use phase may be not the same thing in terms of functional capabilities, but not in structural aspects. This happens because the designer’s view usually is different from the teacher’s view. A designer implements a GLO for the anticipated variability, which may be much larger than the needs of a concrete context of use (e.g., the needs of a teacher). It is the reason why the generated instances are investigated for selection and ordering, in order to let the adjustment to a specific needs of a teacher or a learner. It is also why we introduce the term “qualified set of LO instances” meaning a GLO which is adapted (by a teacher or a learner) to particular needs of the user. The qualified instances are real instances to manage teaching and learning through the creation of sessions of showing this set of instances, as it is identified in the models for teaching and learning (see Figure 4 and 5).

Furthermore, those processes (selection and ordering) may be different for teaching and learning because the aims and tasks, as well as the models for the tasks are slightly different. The framework also outlines different managing strategies. For example, as a result of the execution of one teaching or learning session, the need of using another sequence of instances may be identified. This causes the repetition of the process, which is reflected in Figure 3 with feedback links. The result is re-generation, re-selection, re-ordering, and creation of new sessions for teaching/learning as explained in the next sub-section.

5.3 Integrating mechanism: processes and models

In Figure 4, we present a teaching model that can be used in the context of application of GLOs. Here GLO should be conceived as a qualified sequence of LO instances. Teacher (T) may use some scenario and guidelines, initiates the processes (generation /selection /ordering, as well repeating the processes), manages the set of qualified instances, and provides oral explanations that must be synchronized with what the learner sees in the qualified LO instances. Thus during a session, the knowledge moves from an active teacher towards the usually passive learner. Three modes of knowledge delivery can be used: 1) showing; 2) showing + speaking; 3) showing + speaking + doing.

In this model the focus is given to the showing + speaking mode because doing something (mode 3) is more relevant for learning. A specificity of the model use is a strict dependence of the processes on time constraints.

**Figure 4. Teaching model in the context of using GLOs**

In Figure 5, we present another learning model that can be used in the context of application of GLOs. In this context again, the GLO should be conceived as a qualified sequence of LO instances. Teacher (T) may participate in the learning process or not. All managing activities are initiated by the learner under the predefined guidance and scenario for students. The learner uses some scenario and guidelines, initiates the processes.
(generation / selection / ordering, as well repeating the processes, if needed), manages a set of qualified instances, and performs some doing activities. Knowledge gain is supported by doing activities. The duration of processes are managed by the learner per se. Doing activities may range from the creation of some illustrative examples to gathering of some statistical data or even to performing some experiments (for example, in the case of projects).

How the framework is implemented and works in practice, we demonstrate in our case study.

6 Case study

6.1 A description of the selected topic

Here we consider the array sorting algorithms implemented in various programming languages as LOs. Such LOs could be used in different programming teaching courses to demonstrate the principles and effectiveness of the array sorting algorithms within the e-learning environment independently from a particular programming language. As there are many similarities in the description and implementation of such LOs, they could be described as a single GLO, from which the specific LO instances could be generated at any time.

The Sorting GLO has many variable features: learning objective (principles or effectiveness), sorting algorithm (Bubble Sort, Selection Sort, Insertion Sort, etc.), and demonstration. Demonstration feature includes array size for demonstration, data array population method (random, sorted, sorted in reverse order), sorting order (ascending or descending), and algorithm implementation language (Pascal, C++, Java).

6.2 A characteristic of the selected generative technology

We have selected meta-programming as a generative technology. In general, meta-programming is defined as a manipulation with programs as data. Meta-programming provides mechanisms for writing generic code, i.e. explicitly implementing generalization in the domain. Here we use the heterogeneous meta-programming techniques, which combine two different languages in the same specification: a metalanguage is used at the higher layer of abstraction for representing manipulations on domain language source code, and a domain language is used for representing domain program instances. As typically LO instances are represented in XML (or HTML, XHTML, etc.) format they are actually domain programs. Domain language implements domain commonalities and functionality, while a metalanguage allows developers to specify variations to be implemented in the domain system, and to synthesize customized implementations by composing domain code fragments. Generalization is achieved by the parameterization of differences in different domain program representations, which allows representing domain components with many commonalities in a compact way.

6.3 Characteristics of the designed GLO (with example)

Variability parameters are represented as meta-parameters at a higher abstraction level in the GLO meta-interface, while the commonality-variability relationships are coded at a lower-level within the body of the GLO metaspecification. The role of the meta-interface is to allow an educator to select the parameters of the GLO which reflect different features and variability of the GLO. Selection of the different values of these parameters can be used for LO adjustment to different teaching tasks and student proficiency levels.
The body of GLO is hidden from the end-user (because the technological details of meta-programming are important only for a GLO developer/programmer). By selecting the meta-parameters of GLO and using the meta-language processor, GLO is instantiated, i.e., the specific LO instances are generated. The role of LO instances is to present a specific personalized learning content to the students. These LO instances can be aggregated with other LO instances to make up a module topic (lecture). Therefore, students can access LO instances using an internet browser and can use it for learning module topics as well as for performing individual practice tasks (e.g., to learn by doing).

The values of the variability parameters are described in the GLO meta-interface (see Figure 6). The user (lecturer, teacher) can select different learning objective (e.g., demonstrate the principles of sorting for a particular algorithm, or its effectiveness in terms of operations), select a particular sorting algorithm, select a sorting order (either descending, or ascending), and a programming language for an example of implementation (Pascal, C++, Java). Different combinations of selected values can lead to 432 ($432=2^3\cdot3^2\cdot6\cdot2^3$) different LO instances generated from this single GLO meta-specification.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Options</th>
</tr>
</thead>
<tbody>
<tr>
<td>Learning objective</td>
<td>1- principles, 2- effectiveness</td>
</tr>
<tr>
<td>Algorithm</td>
<td>1- Bubble Sort, 2- Insertion Sort, 3- Selection Sort</td>
</tr>
<tr>
<td>Implementation language</td>
<td>1- Pascal, 2- C++</td>
</tr>
<tr>
<td>Array size</td>
<td>5..10</td>
</tr>
<tr>
<td>Sorting order</td>
<td>1- descending, 2- ascending</td>
</tr>
<tr>
<td>Population method</td>
<td>1- random, 2- sorted, 3- inverse sorted</td>
</tr>
</tbody>
</table>

Figure 6. Meta-interface of GLO

Using the specified interface options, the meta-language processor generates a LO implemented in HTML+Javascript, which can be distributed over Internet. The HTML part of the LO is used for presentation of the natural language description of a sorting algorithm and presentation of its implementation in a specific programming language, while Javascript is used for demonstration of the principles or effectiveness of a specific sorting algorithm.

6.4 Illustrating activities to demonstrate learning by doing

The example of the generated specific sorting algorithm LO is given in Figure 7.

Figure 7. Screenshot of LO instance

LO introduces the student with the description of effectiveness of the Insertion sort algorithm, and demonstrates it in action. The array for sorting is generated after pressing the button “Generate”. And then the sorting effectiveness in terms of performed swap and comparison operations is demonstrated after pressing the button “Insertion sort”. Therefore, this LO can be used not only for introducing the student with the problem of effectiveness of array sorting, but also allows to make a small their own research following the principles of ‘learning-by-doing’ paradigm, e.g., to compare the effectiveness of different algorithms for different array sizes.
and array population types. Such individual learning allows assimilating learner materials better than just seeing, hearing and/or taking notes of what a lecturer says during a lecture or module practice hours.

To better understand ‘learning-by-doing’ in the case of using GLO one should pay attention not only to the specified capabilities for learning (see Figure 6) but also to implementation capabilities (see button “Generate” in Figure 7) more carefully. For example, it is possible to see the intermediate results of sorting, which creates conditions for collecting data (e.g., by changing size of array, see meta-interface) and then doing some research (e.g., for evaluating efficiency of algorithms by multiple execution of the given GLO). “Doing” means re-writing the measures of efficiency by a learner, comparing them for different algorithms, and finally making some conclusions. After receiving knowledge from the given GLO via a learning session and having the core of a sorting program, a learner may perform a real research experiment with less effort. To do so he/she need to extend the core program to the real representation, to generate larger arrays and to repeat the experiment in the environment of the given language.

7 Summary and evaluation of results

We have identified the most essential properties of generative learning objects (GLOs), their role in e-learning and discussed a general framework for creating and using them with the emphasis on the ‘learning by doing’ paradigm. The properties of GLO are: 1) GLO is a higher-level program, actually a generator for delivery of LO instances on demand, if the implementation technology supports that; 2) from the instructor’s and learner’s viewpoint, GLO is a set of related LO instances (or a qualified sequence of LO instances adopted in a particular context of use); 3) if being well-designed, GLO may implement a variety of domain variability aspects (e.g. content-based, pedagogy-based, social-based, etc.) expressed through attributes, their relationships and constraints; variability model should also be conceived as a weak form of ontology, because ontology focuses on knowledge representation in all its aspects and variability focuses on the quantitative characteristics of that knowledge; 4) the well-established specification of a GLO (as a result of the development process managed by a meta-designer) enables to do more with less (e.g., from the learner’s or instructor’s viewpoint), because a) pre-programmed learning aspects are represented explicitly, and b) a generative technology ensures not only productivity but also extends the space for doing (e.g., solving tasks, providing experiments, etc.); 5) In our view, GLOs can make benefit to that learning content best, where a high degree of variability exists and it is to be captured and represented explicitly (under the main focus on quality and productivity issues).

As a result of that, GLOs should be managed adequately in order to integrate them into teaching and learning properly. In the context of the GLOs use, we have discussed the integration problem stating of how GLOs should be used in teaching and learning. A solution of the problem is described through: 1) the presented models, 2) the processes to support the integration (e.g., generation/re-generation of LO instances from a GLO, selection/re-selection of instances, and their sequencing/resequencing) using some integrating mechanism, and 3) the case study.

GLOs bring not only the new capabilities to improve quality and productivity but GLOs also raise new challenges. The development process of GLOs is much more complex. It requires a great deal of analysis in multiple aspects (content, pedagogy, etc.). It requires the introduction of a generative technology as an additional component what makes development processes more complex. It requires a well-established design discipline and tight co-operation among different actors, when designed. Quality assurance is also more difficult. The size of a GLO, which is for using them in real settings, is difficult to identify. A compromise between design complexity and wideness of requirements should be taken into account. Specification of requirements needs some formalism for better complexity management. Experiments with GLOs in real settings are costly; there are little data on that. The maturity level of GLOs is still poor and the research efforts are needed.

The contribution of the paper is as follows: 1) an integrating mechanism and its models describing the way a GLO is integrated into teaching and learning; 2) the enhanced capabilities to implement the ‘learning by doing’ paradigm through the use of generative technology and GLOs.

8 Conclusions

The ideas behind Generative Learning Objects (GLOs) substantially extend the understanding of the concept learning object (LO) per se in e-learning since: 1) GLOs bring the new capabilities to manage variability, 2) to support modifiability and adaptability (e.g., in the context of ‘learning by doing’), 3) to improve quality and productivity, thus enabling to enhance reusability in the domain. But that can be achieved only at a price of the complexity growth and the additional efforts and resources which are needed in the development of GLOs. The field of GLOs opens a wide space for research and for deeper penetration of the generative technology into the e-learning domain. As a finding for the future work, we have identified the importance of analyzing a sequencing problem of LO instances in the context of the use of GLOs, especially in case of large and complex GLOs that contain many instances.
References


