Science of Computer Programming ••• (••••) •••–•••

Contents lists available at ScienceDirect

Science of Computer Programming

www.elsevier.com/locate/scico

Model-driven processes and tools to design robot-based generative learning objects for computer science education

Vytautas Štuikys *, Renata Burbaitė, Kristina Bespalova, Giedrius Ziberkas

Faculty of Informatics, Kaunas University of Technology, Studentų 50, 51368 Kaunas, Lithuania

ARTICLE INFO

Article history:
Received 18 March 2015
Received in revised form 2 March 2016
Accepted 3 March 2016
Available online xxxx

Keywords:
Feature models
Model transformation
Generative learning objects (GLOs)
GLO design tool
Educational robots

ABSTRACT

In this paper, we introduce a methodology to design robot-oriented generative learning objects (GLOs) that are, in fact, heterogeneous meta-programs to teach computer science (CS) topics such as programming. The methodology includes CS learning variability modelling using the feature-based approaches borrowed from the SW engineering domain. Firstly, we define the CS learning domain using the known educational framework TPACK (Technology, Pedagogy And Content Knowledge). By learning variability we mean the attributes of the framework extracted and represented as feature models with multiple values. Therefore, the CS learning variability represents the problem domain. Meta-programming is considered as a solution domain. Both are represented by feature models. The GLO design task is formulated as mapping the problem domain model on the solution domain model. Next, we present the design framework to design GLOs manually or semi-automatically. The multi-level separation of concepts, model representation and transformation forms the conceptual background. Its theoretical background includes: (a) a formal definition of feature-based models; (b) a graph-based and set-based definition of meta-programming concepts; (c) transformation rules to support the model mapping; (d) a computational Abstract State Machine model to define the processes and design tool for developing GLOs. We present the architecture and some characteristics of the tool. The tool enables to improve the GLO design process significantly (in terms of time and quality) and to achieve a higher quality and functionality of GLOs themselves (in terms of the parameter space enlargement for reuse and adaptation). We demonstrate the appropriateness of the methodology in the real teaching setting. In this paper, we present the case study that analyses three robot-oriented GLOs as the higher-level specifications. Then, using the meta-language processor, we are able to produce, from the specifications, the concrete robot control programs on demand automatically and to demonstrate teaching algorithms visually by robot's actions. We evaluate the approach from technological and pedagogical perspectives using the known structural metrics. Also, we indicate the merits and demerits of the approach. The main contribution and originality of the paper is the seamless integration of two known technologies (feature modelling and meta-programming) in designing robot-oriented GLOs and their supporting tools.

* Corresponding author.
E-mail addresses: vytautas.stuikys@ktu.lt (V. Štuikys), renata.burbaite@ktu.lt (R. Burbaitė), kristina.bespalova@ktu.lt (K. Bespalova), giedrius.ziberkas@ktu.lt (G. Ziberkas).

http://dx.doi.org/10.1016/j.scico.2016.03.009
0167-6423/

1. Introduction

There are different approaches to analyze e-learning from various perspectives [1–4]. The *process-based view*, however, seems to be the most relevant to understand the domain in the whole and in the context of computer science (CS) education. At the very abstract level, it is possible to specify and understand the field as a system of interacting components as follows: *pedagogy-driven activities*, *technology-driven processes*, *knowledge transfer channels with actors* involved, a set of *tools* used (they can also be identified as a technology and, when implemented in a concrete setting, it is also known as the *educational environment*), *teaching/learning content* and the *pedagogical/learning outcome* [5]. Among other components, the *teaching/learning content* stands for the central item, because it fuels the remaining components with the basic information to enable the functioning of the whole system.

Typically, the educational content is known as a *learning object* (shortly LO or LOs) in the scientific literature. The term has been introduced by W. Hodgins in the field of e-learning in 1994. There was a *well-founded and far-reaching* intent for that – to resolve the problems related to *systematization, interoperability and reuse of the learning resources*. Though now the role of the concept is well-understood, the understanding of what is meant by the term LO in essence, however, is still poor with various definitions proposed. IEEE, e.g. provides the most general definition stating that LO is "any entity, digital or non-digital, which can be used, re-used or referenced during technology supported learning" [6]. Another definition identifies a LO as the "aggregation of one or more digital assets, incorporating metadata, which represent an educationally meaningful stand-alone unit" [7]. For other definitions, see [3,4,8–10]. Aiming at highlighting some specific features, researchers try to add additional attributes to characterize the item as follows: *reusable* LO [9], *customized* LO [11], *mobile* LO [12], *generic* LO [13,14] to name a few.

Among a variety of LO types, the *generative* LO (shortly GLO, or GLOs) should be considered and analyzed with a specific interest and attention, because now research on GLOs nominates the innovative approaches in e-learning on the whole and in CS education in particular. The concept of GLO results from the contribution of Boyle, Morales et al. [15,16]. For instance, the paper [15] characterizes GLOs as "the next generation learning objects". In terms of the adherence of LOs to the reusability issues, the introduction of this concept means much more than the conventional LO. In fact, the pioneers of GLOs send the obvious message to the extremely large LO research community to move from the *component-based reuse model* to the *generative reuse model*. Therefore, a GLO implements the generative reuse model using the adequate technology. Again, there is no uniform definition of GLO. For instance, the Center for Excellence in the design, development and use of LOs in UK (shortly, RLO-CETL) defines GLO as "an *articulated* and *executable* learning design that produces a class of learning objects" (http://www.rlo-cetl.ac.uk/whatwedo/glos/whatareglos.php). We will present another definition later.

The articulation in the given definition is understood in two different ways: (1) as human-understandable explicit or implicit decisions involved in design for learning and (2) these decisions can be executed by computer software to produce LOs based on the design. In practice, i.e. when the GLO Authoring tool GLO Maker (http://www.glomaker.org) is used, the pedagogical designs are represented explicitly as the *'plug-in' patterns*. The tool is used to create specific LOs based on the chosen pattern. Each of these LOs created in this way can be re-purposed by the local users, using the same tool, to adapt the resources to their needs and preferences. Then all the LOs so created (or adapted) run as stand-alone Web based LOs. This approach has been borrowed from the systemic grammar [17]; however, it can be also seen (in the use sense) as the template-based approach. In terms of software reuse, the template-based approach is treated as the simplest generative technology [18].

In the course of the GLO concept evolution, other approaches to implement the model were introduced and applied. In [19], for instance, GLOs are implemented using heterogeneous meta-programming (He MPG) as a generative technology. The latter is defined as the programming paradigm that uses at least two languages to represent the GLOs in the same specification. The first is the meta-language. Its role is to describe the generative (i.e. generalization) aspects through parameterization to implement the generalization. The second is the target language. Its role is to express the basic functionality of the target program to be taught or learned. Therefore, the target language is also the teaching language in which, for example, the data structure and algorithms (or their parts) are expressed. Any target language can be selected in designing GLO specifications (for instance, according to the curricular requirements), because the He MPG paradigm is independent upon the use of the target language. Such GLOs represent the pre-programmed design for learning. This design, in fact, realizes the concept of the so-called learning variability [5]. Therefore, the meta-programming-based GLO is defined as follows.

GLO of this kind is the *high-level executable* (*i.e. pre-programmed*) *specification* that specifies the *learning variability* and a particular variant from the specification is derived on the user's (learner's or teacher's) demand as a LO automatically using the meta-language processor. The latter along with the GLO specification is *the learning content* (*i.e. LO*) *generator*. The learning variability is the compound that includes the interrelated parts such as the pedagogy-related aspects (e.g. explicit teaching aim, teaching model, etc.), social-related aspects (e.g. learner's profile, level of previous knowledge, etc.), technological aspects (e.g. characteristics of educational tools and environment, etc.) and pure content aspects (e.g. in case of CS education, data structures, algorithms or their parts, etc.). The basic idea of the approach is that first we are able to express *the value* of each element of the component *explicitly and uniformly* through features and their relationships using feature-based modelling concepts [20]. Then, while creating meta-specifications based on feature model transformations [21],[1] the

[1] Originally the feature notion was introduced in software engineering by Kang et al. in 1990 and in e-learning by Dodero et al. in 2007.

Please cite this article in press as: V. Štuikys et al., Model-driven processes and tools to design robot-based generative learning objects for computer science education, Sci. Comput. Program. (2016), http://dx.doi.org/10.1016/j.scico.2016.03.009