Computer Languages, Systems & Structures 43 (2015) 116–138

Contents lists available at ScienceDirect

Computer Languages, Systems & Structures

journal homepage: www.elsevier.com/locate/cl

Grammar-based model transformations: Definition, execution, and quality properties

Galina Besova [*], Dominik Steenken, Heike Wehrheim

Department of Computer Science, University of Paderborn, Warburgerstr, 100, 33098 Paderborn, Germany

ARTICLE INFO

Article history:
Received 28 November 2014
Received in revised form
3 April 2015
Accepted 10 May 2015
Available online 19 May 2015

Keywords:
Model transformations
Graph grammars
Quality properties

ABSTRACT

Model transformation is a key concept in model-driven software engineering. The definition of model transformations is usually based on meta-models describing the abstract syntax of languages. While meta-models are thereby able to abstract from superfluous details of concrete syntax, they often loose *structural* information inherent in languages, like information on model elements always occurring together in particular shapes. As a consequence, model transformations cannot naturally re-use language structures, thus leading to unnecessary complexity in their development as well as in quality assurance.

In this paper, we propose a new approach to model transformation development which allows to simplify the developed transformations and improve their quality via the exploitation of the languages' structures. The approach is based on context-free *graph grammars* and transformations defined by *pairing* productions of source and target grammars. We show that such transformations have important properties: they *terminate* and are *sound*, *complete*, and *deterministic*.

1. Introduction

Model transformations are key to model driven engineering (MDE). Surveys on model transformations [1,2] show their expanding application areas: model translation, model composition, refinement, abstraction, and others.

In an MDE setting, the syntax of models is given in terms of *meta-models* which themselves conform to their own meta-models (e.g., MOF [3]). Meta-models define the *abstract syntax* of languages and omit the details of concrete syntax, like keywords and ordering of elements. While meta-models describe model elements and their direct relations, they often fall short of describing model elements always occurring together in particular shapes/structures (e.g., decision/merge blocks in UML Activity diagrams [4]). In some cases, meta-models are enriched with OCL [5] constraints to enforce such shapes in models. Model transformations are defined on meta-models and thus operate on abstract syntax.

In contrast to MDE, traditional approaches to language definition (and translation) define languages by *grammars*, often given in an Extended Backus–Naur Form (EBNF) [6]. These translation techniques operate on concrete syntax. While the details of concrete syntax are in general unimportant (and thus make translation definition unnecessarily confusing), the meta *structural* information contained in the grammar-based language definitions is highly useful for defining translations.

[*] Corresponding author.
E-mail addresses: besova@mail.upb.de (G. Besova), dominik@mail.upb.de (D. Steenken), wehrheim@mail.upb.de (H. Wehrheim).

http://dx.doi.org/10.1016/j.cl.2015.05.003
1477-8424/© 2015 Elsevier Ltd. All rights reserved.

The productions of the grammars define the structures available in the languages, and by relating productions of grammars (as done in syntax-directed translation [7]) we can naturally specify how languages' structures are mapped onto each other.

An ideal approach for model transformation should thus combine these two approaches, taking the best of both: use language definitions with the abstract syntax of meta-models and the structures of grammars, and build model transformations on these definitions. An early approach following this idea, although not in the area of model transformations and not with meta-models but with graphs, is the one proposed by Pratt [8]. Pratt defines *pair grammars* as a way of relating the grammars of two languages, thus obtaining a natural structure-oriented way of relating languages and building translations between them.

The objective of this paper is to bring the pair grammar based translation to the world of MDE model transformations, lifting it to the level of abstract syntax while preserving its advantages. It also presents several ideas to extend this approach to cover a broader variety of model transformations, and provides formalization of both the basic approach and of these extensions. This paper is an extended version of our earlier work [9] with additional sections on the formal aspects of the definition, execution, and quality properties of the transformations defined using the proposed approach.

Fig. 1 gives an overview of our approach. The transformations we focus on are model-to-model transformations. Our models are given in abstract syntax and are generated by grammars. For this generation purpose we use a type of context-free graph grammars – *hyperedge replacement graph grammars* [10] – typed and constrained by meta-models. Grammar-based rules – like pair grammars – relate productions of the source with those of the target grammar. We refer to the resulting transformations as *grammar-based model transformations* (GBMTs). GBMTs are executed on *derivation trees*: given a source model M_s, its derivation tree in the source grammar is obtained by parsing, and used by the GBMT to produce a derivation tree in the target grammar and the corresponding target model M_t.

We exemplify our approach on a transformation from activity diagrams to the *Communicating Sequential Processes* (CSP) algebra [11]. On this example we see how the definition of transformation rules between language structures (e.g., decision/merge and if-then-else blocks) can be simplified using their explicit definition in productions in our grammar-based approach. For comparison, in Section 6, we provide a typical more complex rule created for the same purpose using existing state-of-the-art methods (here ATL). Furthermore, in Section 7, we highlight further benefits of our grammar-based approach. It is concerned with automatic implicit connection of the results of grammar-based transformation rules, compared to the more complex explicit, and often imperative, one in the existing approaches in MDE.

We also show important qualities of the transformations developed with our approach – *termination*, *soundness*, *completeness*, and *determinism*. Showing these quality properties for a transformation described using current meta-model based state-of-the-art techniques is usually hard [12], as discussed in Section 8.

First, in Section 2 we give background on grammar-based language definition and show our source and target grammars. Then, we introduce our grammar-based approach using an example transformation in Section 3 and provide its formalization in Section 4. In Section 5, we consider the quality of the transformations developed using our approach. In Section 6, we evaluate it in comparison with the most closely related approaches in the MDE context. In Section 7, we evaluate the application of our grammar-based approach to a further example. Finally, we survey related work in Section 8 and conclude in Section 9.

2. Background

There are two fundamentally different ways of specifying the syntax of a given language: with (context-free) grammars, and with meta-models. Our approach is built on grammars generating instances of meta-models, i.e., graphs. In the following, we introduce the main concepts of grammar-based language definition and show how they can be lifted to graph-based languages, enabling grammar-based definition of modeling languages and model transformations utilizing these definitions. We show how our example of modeling languages for activity diagrams and CSP can be described using grammars. Finally, we introduce the transformation example used later to demonstrate our approach.

2.1. Grammar-based syntax definition

In their original usage, grammars define languages of strings via a set of generative rules. We briefly review main definitions of grammars for string languages from [7], since the graph grammars we use are their natural extension.

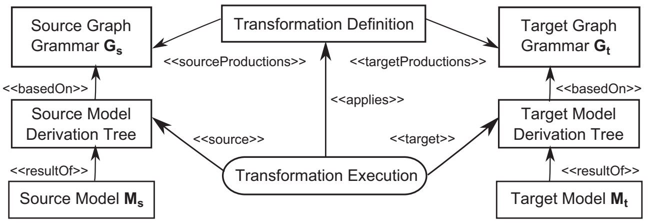

Fig. 1. Overview of our approach.