Information and Software Technology 62 (2015) 124–142

Contents lists available at ScienceDirect

Information and Software Technology

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

# Assessing the use of slicing-based visualizing techniques on the understanding of large metamodels

Arnaud Blouin [a,*], Naouel Moha [b], Benoit Baudry [c], Houari Sahraoui [d], Jean-Marc Jézéquel [e]

[a] INSA Rennes, IRISA/Inria, Diverse Team, Rennes, France
[b] University of Québec at Montréal, Montréal, Canada
[c] Inria, IRISA/Inria, Diverse Team, Rennes, France
[d] University of Montréal, GEODES Group, Montréal, Canada
[e] University of Rennes 1, IRISA/Inria, Diverse Team, Rennes, France

ARTICLE INFO

*Article history:*
Received 1 November 2014
Received in revised form 11 February 2015
Accepted 12 February 2015
Available online 25 February 2015

*Keywords:*
Model-Driven Engineering
Metamodel
Class diagram
Visualization
Human–computer interaction
Model slicing

ABSTRACT

*Context:* Metamodels are cornerstones of various metamodeling activities. Such activities consist of, for instance, transforming models into code or comparing metamodels. These activities thus require a good understanding of a metamodel and/or its parts. Current metamodel editing tools are based on standard interactive visualization features, such as physical zooms.
*Objective:* However, as soon as metamodels become large, navigating through large metamodels becomes a tedious task that hinders their understanding. So, a real need to support metamodel comprehension appears.
*Method:* In this work we promote the use of model slicing techniques to build interactive visualization tools for metamodels. Model slicing is a model comprehension technique inspired by program slicing. We show how the use of `Kompren`, a domain-specific language for defining model slicers, can ease the development of such interactive visualization features.
*Results:* We specifically make four main contributions. First, the proposed interactive visualization techniques permit users to focus on metamodel elements of interest, which aims at improving the understandability. Second, these proposed techniques are developed based on model slicing, a model comprehension technique that involves extracting a subset of model elements of interest. Third, we develop a metamodel visualizer, called `Explen`, embedding the proposed interactive visualization techniques. Fourth, we conducted experiments. showing that `Explen` significantly outperforms `EcoreTools`, in terms of time, correctness, and navigation effort, on metamodeling tasks.
*Conclusion:* The results of the experiments, in favor of `Explen`, show that improving metamodel understanding can be done using slicing-based interactive navigation features.


## 1. Introduction

The fundamental idea of Model-Driven Engineering (MDE) is to consider models as first-class entities. A model conforms to a metamodel that describes the concepts and relationships of a given domain. Metamodels, usually represented graphically as class diagrams, are thus cornerstones of various metamodeling activities. Such activities consist of, for instance, transforming models into code, creating editing tools for a metamodel, or comparing metamodels. These activities thus require a good understanding of a metamodel and/or its parts. Understanding metamodels mainly consists of understanding the relations between classes of interest by navigating between them through their inheritance or reference relations. The current mainstream metamodel editors, such as `EcoreTools` provided by the Eclipse Modeling Framework (EMF),[1] however, only offer basic interactive features to navigate through metamodels (physical zoom, scroll bars, etc.). Physical zooms are used to change the size of metamodels' elements, scroll bars are used to navigate from one class to another one, and several filters permit hiding classes or relations. Although MDE promotes the separation of concerns that should limit the size of metamodels by decomposing them into small ones, empirical evidence shows

* Corresponding author.
*E-mail addresses:* ablouin@irisa.fr (A. Blouin), moha.naouel@uqam.ca (N. Moha), bbaudry@inria.fr (B. Baudry), sahraoui@iro.umontreal.ca (H. Sahraoui), jezequel@irisa.fr (J.-M. Jézéquel).

[1] http://www.eclipse.org/modeling/emf/.

http://dx.doi.org/10.1016/j.infsof.2015.02.007
0950-5849/

that many of them are large. An empirical study we conducted on 3462 well-formed Ecore domain metamodels we gathered from the github platform highlighted that: 82% of the studied metamodels are composed of a single package, the mandatory root package; 15%, *i.e.* 508, of these metamodels are composed of 40 classes or more. As soon as metamodels become large, understanding and manipulating metamodels becomes a tedious task using these basic interactive features. For instance, Fig. 1 is an overview of the UML metamodel [1] obtained using the physical zoom of `EcoreTools`. Many classes are gathered and reduced so that identifying one class or its relations with other ones becomes awkward. As noticed by Zhao et al., "*while node-link diagrams show nesting structure very clearly, they use screen space inefficiently, and do not scale well to large datasets*" [2].

When modelers are interested only in a specific part of a metamodel, they may want to focus on it by, for instance, hiding the rest of the metamodel. For instance, for the visualization of a metamodel, a modeler may be interested in semantic relationships between classes such as: the inheritance tree of a given class; the classes linked by a composition reference to a given class. As motivated by Fondement et al., "*by indicating formally the subset of the metamodel that is actually covered, a tool could be made more precise regarding handled model*" [3]. With the current editors, modelers are forced to manually and astutely combine sequences of filtering and navigation primitive operations to rebuild these parts of interest. This manual exploration task may be time-consuming and error-prone. So, a real need to support metamodel comprehension appears.

Visualization techniques are broadly used in software engineering and have proven their usefulness for software comprehension and in particular, interactive visualization that provides meaningful navigation capabilities [4]. Gračanin et al. summarized the benefits in terms of comprehension brought by software visualization to different domains such as software evolution, software security, and data mining [5]. Previous works on UML class diagrams highlight the research interest on improving the understanding of class diagrams [6–8]. These works mainly focus on proposing new algorithms and methods for minimizing relations crossing [9–11] or guidelines for drawing class diagrams [12,13]. Other research works proposed to represent class models differently than using class diagrams [14] or in 3D [15–17]. In this work we focus on metamodeling tasks that modelers perform while handling metamodels. More precisely, we consider how to produce interactive visualization features dedicated to metamodels rather than the rendering of metamodels. We also keep the focus on the class diagram representation promoted and widely-used within the MDE community. We specifically propose four main contributions. First, we propose interactive visualization techniques that permit users to focus on metamodel elements of interest. These techniques aim at improving the understandability of metamodels. Second, these proposed techniques are developed based on model slicing [18,19]. Model slicing is a model comprehension technique inspired by program slicing [20]. The process of model slicing involves extracting a subset of model elements of interest. We show how the use of `Kompren`, a domain-specific language for defining model slicers [18,19], can ease the development of such interactive visualization features. Third, we develop a metamodel visualizer, called `Explen`, embedding the proposed interactive visualization techniques. Fourth, we conducted an empirical study to measure the possible benefits, in terms of time, correctness, and navigation effort, when performing metamodeling tasks using `Explen` compared to the mainstream metamodeling tool `EcoreTools`. This study exhibits significant positive results for `Explen` regarding both time (30% better in favor of `Explen`), correctness (22% better in favor of `Explen`), and navigation effort (50% better in favor of `Explen`). This work is the first step towards generalizing the proposal to any kind of models represented with a graphical syntax. It aims at validating the benefits of the proposal on metamodels to then, in future work, consider models in general.

This paper extends our work published at VISSOFT 2014 (*New Ideas or Emerging Results* Track) [21] with an empirical study, an exhaustive study of the related work, and more details explaining the proposed interactive visualization features.

The paper is organized as follows. Section 2 motivates this work by presenting a scenario that highlights the need for integrating interactive visualization features within graphical modeling tools. Section 3 describes how model slicing can be leveraged to develop interactive visualization techniques for the visualization of large metamodels. Section 4 details the experimental design. Section 5 analyses and comments the results of the conducted experiments. The paper ends with the related work in Section 6 and the conclusion in Section 7.

## 2. Motivating scenario

We motivate the need for integrating interactive visualization features within graphical modeling tools based on the following common scenario.

**Scenario.** A modeler has to write a model transformation that generates Java code from UML 2.0 models. The modeler has already a rough idea of the main classes required for the transformation: *Association*, *Class*, *Package*, *Parameter*, *Property*, and *Operation*. However, before writing the transformation, the modeler needs to have a clear and precise understanding of how these classes are organized within the UML metamodel and identify the properties and operations required for the transformation. To acquire this

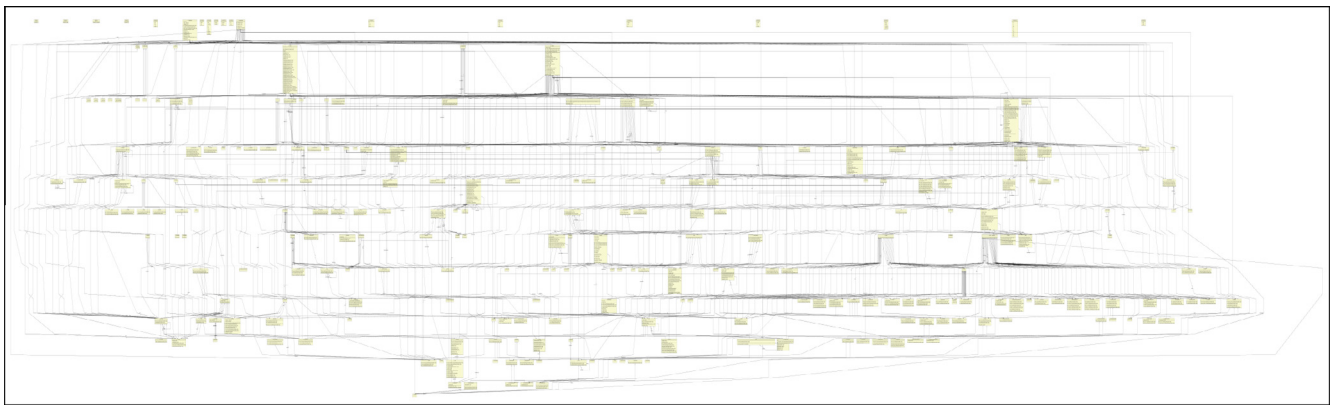

**Fig. 1.** Bird view of the UML metamodel using `EcoreTools` (246 classes and 769 relationships).