Jan Jürjens, Kurt Schneider (Hrsg.): Software Engineering 2017,
Lecture Notes in Informatics (LNI), Gesellschaft für Informatik, Bonn 2017 137

# GECO: A Generator Composition Approach for Aspect-Oriented DSLs

Reiner Jung[1] Robert Heinrich[2] Wilhelm Hasselbring[3]

**Keywords:** Megamodel Patterns, Generator Construction, Generator Evolution

## 1 Introduction

Models play a central role in the specification of software systems. As models cannot be executed directly, model and code generators are used to transform them into implementation artifacts. These generators face challenges caused by complexity and tight coupling of generator implementations, particularly when multiple metamodels are involved. As a consequence maintenance, evolution and reuse of generators is expensive and error-prone.

We address these challenges with a two fold GEnerator COmposition approach (GECO [JHH16]), which subdivides generators in fragments and modules: (1) fragments are combined utilizing megamodel patterns. These patterns are based on the relationship between base and aspect metamodel.(2) fragments are modularized along transformation aspects, such as model navigation, and metamodel semantics.

We evaluated GECO with two case studies from different domains, and assessed the generators with modularity and complexity metrics, covering architecture and method level.

## 2 Approach

GECO is technology agnostic, covers both code and model generators, and is applicable to to Aspect-Oriented Modeling [KAK09] and Multi-View Modeling [ASB10] approaches.

In GECO, generators are modularized on two levels. First they are divided into smaller generators, called fragments, and then subdivided into modules. Each fragment is defined with only one source and target metamodel, and can often be realized with one transformation. As metamodels may not be self-contained and may cover multiple views and aspects, fragments can be designed for only a partition of a source metamodel, especially for partitions that fulfill the criteria of an aspect or base metamodel [Ju14, Ju16, St16]. This implies that it is not necessary to have de facto multiple metamodels to developed with GECO. It is sufficient to be able to partition the metamodel along the relationships of base and aspect models, and independent and dependent views, respectively.

[1] Kiel University, Christian-Albrechts-Platz 4, 24118 Kiel, reiner.jung@email.uni-kiel.de
[2] Karlsruhe Institute of Technology, Am Fasanengarten 5, 76131 Karlsruhe, robert.heinrich@kit.edu
[3] Kiel University, Christian-Albrechts-Platz 4, 24118 Kiel, hasselbring@email.uni-kiel.de

## 3 Conclusions

The GECO approach provides the means to modularize generators which supports their construction and evolution.

We evaluated GECO in two case studies representing information systems and embedded control systems. In both cases GECO help to modularize generators and the GECO-based generators showed lower complexity compared pre-existing generator implementations. Supplementary, GECO provides reusable modules for generator development supported by a DSL. More details on the evaluation can be found in [JHH16] and [Ju16].

Our future work will explore two primary avenues of investigation. First, we will evolve the generators for the information system case study in context of the iObserve project [Ha13]. Second, we want to investigate generator performance and memory consumptions as these are two important limiting factors of code generators in general.

**Acknowledgement** This work was supported by the DFG (German Research Foundation) under the priority program SPP 1593: Design For Future – Managed Software Evolution (grants HA 2038/4-1, RE 1674/7-1) and the Helmholtz Association of German Research Centers.

## References

[ASB10] Atkinson, Colin; Stoll, Dietmar; Bostan, Philipp: Orthographic Software Modeling: A Practical Approach to View-Based Development. In: Evaluation of Novel Approaches to Software Engineering, volume 69 of CCIS, pp. 206–219. Springer, 2010.

[Ha13] Hasselbring, Wilhelm; Heinrich, Robert; Jung, Reiner; Metzger, Andreas; Pohl, Klaus; Reussner, Ralf; Schmieders, Eric: iObserve: Integrated Observation and Modeling Techniques to Support Adaptation and Evolution of Software Systems. Research report, Kiel University, Kiel, Germany, Oct 2013.

[JHH16] Jung, Reiner; Heinrich, Robert; Hasselbring, Wilhelm: GECO: A Generator Composition Approach for Aspect-Oriented DSLs. In (Van Gorp, Pieter; Engels, Gregor, eds): ICMT, Vienna, Austria, July 4-5, 2016, Proc. Springer, pp. 141–156, 2016.

[Ju14] Jung, Reiner; Heinrich, Robert; Schmieders, Eric; Strittmatter, Misha; Hasselbring, Wilhelm: A Method for Aspect-oriented Meta-Model Evolution. In: 2. VAO Workshop. ACM, pp. 19:19–19:22, 2014.

[Ju16] Jung, Reiner: Generator-Composition for Aspect-Oriented Domain-Specific Languages. Dissertation, Faculty of Engineering, Kiel University, August 2016.

[KAK09] Kienzle, Jörg; Abed, Wisam Al; Klein, Jacques: Aspect-oriented multi-view modeling. In: Proceedings of the 8th ACM International Conference on Aspect-oriented Software Development. ACM, pp. 87–98, 2009.

[St16] Strittmatter, Misha; Hinkel, Georg; Langhammer, Michael; Jung, Reiner; Heinrich, Robert: Challenges in the Evolution of Metamodels: Smells and Anti-Patterns of a Historically-Grown Metamodel. In (Mayerhofer, Tanja et al., ed.): Proc. of the 10th Workshop on Models and Evolution. volume 1706. CEUR, pp. 30–39, Oct 2016.