Henrik Leopold, Hend*erik* A. Proper (Hrsg.): EMISA 2022,
Gesellschaft für Informatik, Bonn 2022 1

Retrofitting Generative Aspects in Existing Applications

Arkadii Gerasimov[1], Judith Michael[1] *, Imke Nachmann[1], Lukas Netz[1], Bernhard Rumpe[1], Simon Varga[1]

Abstract: Incorporating model-based approaches and code synthesis into the engineering process of information systems is a challenging task in practice as brownfield methods and successful role models are often found to be lacking. We have investigated how to integrate generative aspects in already existing applications and propose a methodology encompassing all engineering process steps from problem analysis to system operation and maintenance. We have applied the methodology to a real-world enterprise information system with more than 500,000 lines of code and provide developers with detailed insights on the practical application. The findings presented in this current research talk appeared in 2021 as Ä Methodology for Retrofitting Generative Aspects in Existing Applications"within the Journal of Object Technology.

Keywords: Methodology; Model-Based Software Engineering; Code Synthesis; Brownfield; Information System; Problem Decomposition.

1 Current Research Talk

Motivation and Approach. The use of model-based techniques and automation, such as code synthesis, reduce the conceptual gap and allow for faster software engineering processes [Vö13]. In practice, their use is a challenging task due to already existing applications which continuously increase in size and complexity [FR07]. Therefore, research has to adapt its model-based engineering methodologies towards brownfield. They have to allow incremental incorporation of models and continuously evolving code generators producing source code that can replace hand-written code. Within the journal article Ä Methodology for Retrofitting Generative Aspects in Existing Applications-[Dr21], we have addressed the research question *how one could retrofit generative aspects in already existing enterprise information systems*. We propose a model-based methodology for the incremental reconstruction of code parts of existing applications in the brownfield. The methodology was applied to a software for the financial management and controlling of university chairs developed in the MaCoCo project [Ad18].

Methodology and Application. Our method [Dr21] includes three phases, namely (1) problem analysis and decomposition, (2) domain-specific language engineering, and (3) application engineering and operation which are further divided into 16 activities including

[1] Software Engineering, RWTH Aachen University, Germany
{gerasimov,michael,nachmann,netz,rumpe,varga}@se-rwth.de
* corresponding author

CC BY SA

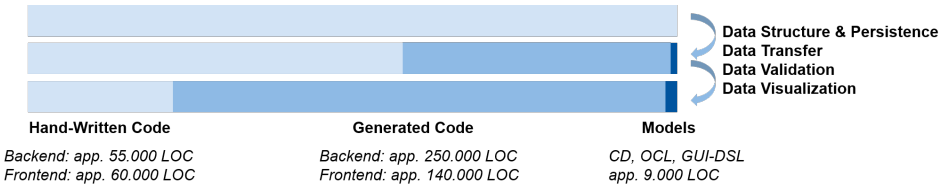

Fig. 1: Iterative development of MaCoCo using the proposed methodology

optional ones. Phase 1 starts with a hand-written software application which is further analyzed and decomposed into smaller problem aspects such as data validation, persistence, or GUI. In phase 2, activities for domain-specific language engineering are applied. This includes identifying relevant language families and time-phases to apply models, choosing a concrete language, and optional language engineering. Finally, in phase 3, activities for application engineering and operation of the software system are applied, e.g., generator and model engineering, hand-written additions, testing, migration, and continuous deployment.

Fig. 1 shows how the iterative development of MaCoCo using our methodology has changed the code structure. It has shifted from an entirely hand-written prototype to an application where 78% of the code is generated from 9.000 LOC in models. Within [Dr21], we discuss this transformation process and our decisions and provide practical insights for developers who consider applying the methodology to their existing information systems, e.g., how to divide the problem aspects, when to choose which language family, when to reuse existing languages for better maintainability or create a new one, insights on generator reuse, challenges with data migration, and the need for automated testing.

Conclusion. The proposed methodology facilitates the application of model-based techniques and code generation to existing systems. Complex and large software systems can be divided into smaller aspects and hand-written parts are iteratively replaced by models and generated code.

Bibliography

[Ad18] Adam, Kai; Netz, Lukas; Varga, Simon; Michael, Judith; Rumpe, Bernhard; Heuser, Patricia; Letmathe, Peter: Model-Based Generation of Enterprise Information Systems. In: Enterprise Modeling and Information Systems Architectures (EMISA'18). volume 2097 of CEUR Workshop Proceedings. CEUR-WS.org, pp. 75–79, 2018.

[Dr21] Drave, Imke; Gerasimov, Akradii; Michael, Judith; Netz, Lukas; Rumpe, Bernhard; Varga, Simon: A Methodology for Retrofitting Generative Aspects in Existing Applications. Journal of Object Technology, 20:1–24, November 2021.

[FR07] France, Robert; Rumpe, Bernhard: Model-driven Development of Complex Software: A Research Roadmap. Future of Software Engineering (FOSE '07), pp. 37–54, May 2007.

[Vö13] Völter, Markus; Stahl, Thomas; Bettin, Jorn; Haase, Arno; Helsen, Simon: Model-Driven Software Development: Technology, Engineering, Management. Wiley Software Patterns Series. Wiley, 2013.