Dominik Bork, Dimitris Karagiannis, Heinrich C. Mayr (Hrsg.): Modellierung 2020,
Lecture Notes in Informatics (LNI), Gesellschaft für Informatik, Bonn 2020 193

Models as Programs

A Tutorial

Bernhard Thalheim[1]

Abstract: Models are one of the main instruments for system development in computer science and engineering. So far models have been used for system description and system prescription, i.e. essentially as a blueprint for development. Models might however become programs for themselves. This approach allows to claim that models will become the kernel element of true fifth generation programming.

Keywords: models as programs; model-based programming; modelling

1 Towards True Fifth Generation Programming

Programming became more and more comfortable with development of third and fourth generation programming languages. Although the fifth generation project did not achieve its goals, the necessity for more comfortability is still challenging. This tutorial delineates the path towards *next generation programming* [JT19] based on *literate modelling* with model suites[2] that generalises model-driven development and conceptual-model programming [TSF19].

2 Tutorial Outline and Schedule

2.1 Models as next generation programs for everybody

Programming has become a technique for everybody, especially for non-computer scientists. Programs became an essential part of modern infrastructure. Programming is nowadays a socio-material practice in most disciplines of science and engineering. Solution development for real life complex systems becomes however an obstacle course due the huge variety of languages and frameworks used, due to impedance mismatches among libraries, due to conflicting environments, due to vanishing programming expert knowledge and culture,

[1] Department of Computer Science, Christian-Albrechts University of Kiel, 24098 Kiel, Germany, thalheim@is.informatik.uni-kiel.de

[2] A *model suite* consists of a coherent collection of explicitly associated models. A model in a model suite is used for different purposes such as communication, documentation, conceptualisation, construction, analysis, design, explanation, and modernisation. The model suite can be used *as a program of next generation* and will be mapped to programs in host languages of fourth or third generation. So, we claim that *models will become programs of next generation programming*.

due to novel and only partially understood paradigms such as componentisation and app programming, due to the inherent tremendous complexity, due to programming in the large and programming in the web, and due to legacy and integration problems. Our goal is very ambitious. This tutorial addresses the challenges by developing foundations and technologies for next generation programming and testing them in three central areas of computer science. Models will become programs of next generation programming. We developed and implemented literate modelling with model suites as generalisations of model-driven development and of conceptual-model programming.

2.2 Model suite description language

The model language consists of an associated bundle of languages for model elements that users may modify depending on their needs or simply reuse them as already established subcultures. These elements are different from ordinary programs because they are essentially declarative rather than imperative. Similar to UML stereotypes, MaP model class and model style languages are ready for application, can be extended, combined, and adapted. Users don't have to work on the details of the models as programs (MaP). The system takes over the integration and composition work as it deduces the consequences of the model.

2.3 Technologies, techniques, methodologies, and modelling moulds

The development of models in a model suite is based on a model library with models that can be used as an inherited or initial model for systematic composition of the model suite. The approach to model construction is canonised on the basis of methodologies and modelling moulds which systematically combine known and novel techniques and technologies for model development. Modelling moulds enable the modeller to reuse systematics and theories that have been successfully deployed in the past. They enable us to start with application space models, with deep models, with generic models, and with reference models without explicit reinvestigation of these models. The explicit agreement on a given mould eases, enables, and supports an economic development process.

2.4 Environment for an extension towards modelware as next generation literate modelling

Our approach aims at development of a general infrastructure for treatment of models as programs. This infrastructure includes also standardised solutions that can be reused in other applications. These solutions are based on application space models and deep models that are typically less volatile than normal models. Therefore, the library allows quick and well-based modelling by non-specialists which may concentrate on development of normal models instead of developing the entire holistic model. They may accept the library models as a basis and then use generic and reference models as a starting point for normal model development. The model suite is also transformed to programs in programming languages of third or fourth generations. This approach disentangles modellers from programming and allows them to concentrate on the model development. The model is then the product. The transformed program is of a higher quality and more liable.

2.5 Compiler-compiler approach to model realisation for models as programs

The modelling infra-structure is an essential element for realisation of model suites and for treatment of models as programs of next generation programming. The compiler-compiler approach is enhanced by the layered handling of models. This generation is the basis for language and platform independence of the models themselves. Compilation allows the integration of standards. Model suite libraries become then the kernel for modelware. Models become directly executable.

Literaturverzeichnis

[BT08] Börger, Egon; Thalheim, Bernhard: A method for verifiable and validatable business process modeling. In: Advances in Software Engineering, S. 59–115. Springer, 2008.

[JT19] Jaakkola, Hannu; Thalheim, Bernhard: Models as programs: The envisioned and principal key to true fifth generation programming. In: In Proc. 29'th EJC, S. 170. Lappeenranta, Finland, 2019.

[KT16] Kramer, Frank; Thalheim, Bernhard: Holistic Conceptual and Logical Database Structure Modeling with ADOxx. In: Domain-Specific Conceptual Modeling, S. 269–290. Springer, 2016.

[MT19] Molnár, András J.; Thalheim, Bernhard: Usage Models Mapped to Programs. In: New Trends in Databases and Information Systems. Springer International Publishing, Cham, S. 163–175, 2019.

[TSF19] Thalheim, Bernhard; Sotnikov, Alexander; Fiodorov, Igor: Models: The Main Tool of True Fifth Generation Programming. In: Selected Papers of the XXII International Conference Ënterprise Engineering and Knowledge Management (EEKM 2019), S. 161–170. 2019.