Software and Systems Modeling (2021) 20:1333–1334
https://doi.org/10.1007/s10270-021-00932-8

EDITORIAL

The wild-west of modeling (Revisited)

Jeff Gray[1] · Bernhard Rumpe[2]

Published online: 3 October 2021

Modeling is a creative and intellectually focused activity that has a long history and tradition. Ancient Greek philosophers discussed "what is original" and "how to discern the essential nature of a thing" using models. They endeavored to understand the physical world, the biological world, humanity, human medical conditions, and so on by reducing concrete and specific concepts into simpler analogies and rules. One could even argue that the very first models were cave drawings, which show hunting scenes (such as the one on SoSyM's cover!) to describe the process of hunting successfully for teaching purposes. Cave drawings therefore have a purpose and fulfill the general criteria for being a model. The actual word "modeling" was used already in the twelfth century in Italy, when 1:10 miniature and wooden buildings from churches were called "models" and were used to represent a newly constructed building for stakeholder and developer discussions (and potentially to help raise the money needed for construction) before actually creating the real physical structure.

Modeling is one of the key supports for scientific and engineering disciplines. However, the use of explicit and precisely defined modeling languages is relatively new and has developed primarily in the context of software support for the modeling activity. Interaction with machines has enforced precise syntactic forms; that is, the machine clearly accepts or rejects a model, before any computation is performed or transformation occurs into some other kind of model, executable program, or test infrastructure. A precise semantics requires definition through the behavior of a machine.

In software development, the main form of a computational expression is through a programming language. The UML was originally designed as a high-level abstract modeling language to define architecture, capture requirements, effectively model tests, and allow tools to trace high-level requirements down to code. The UML has made progress toward these goals, but it is far less used as originally anticipated. There are various reasons for this, including the observable semantic gap between high-level programming languages and their sophisticated libraries, in comparison with abstract modeling concepts. The immaturity of tool support for UML in its early history is also a well-known limitation that hampered the full achievement of the original goals of the UML.

However, the situation is somewhat different when we consider non-software contexts. In particular, in systems engineering the SysML is defined with a new version, as well as a large set of domain-specific languages (DSLs) that are used in various subdomains of systems engineering. It is our subjective experience that in these domains the use of explicit modeling languages is becoming much more prominent. It may even be that the term "MBSE" in the future does not stand for "Model-Based Software Engineering," but for "Model-Based Systems Engineering".

Several decades ago, when MBSE was beginning to emerge, software companies were created with the core purpose of selling tool infrastructure. For a variety of reasons, the commercialization did not reach the expectations. Independent companies, such as GentleWare™ and Rational™, ceased to exist on their own and were often bought out and merged into divisions of larger corporate entities. The early phase of the software modeling wild-west gold digger period did not survive at scale, but the modeling community gained much experience on how to develop languages and tools, with a specific focus on what the users really want and need.

The "model-based systems engineering" community may be following the same wild-west path of the early MBSE period: Systems modeling tooling companies are likely to be purchased because they have sophisticated and highly

✉ Bernhard Rumpe
bernhard.rumpe@sosym.org

Jeff Gray
jeff.gray@sosym.org

[1] University of Alabama, Tuscaloosa, AL, USA

[2] RWTH Aachen University, Aachen, Germany

customizable editors for SysML and other specific DSLs. SysML 2.x may create a rush for the next big set of toolings. New approaches may emerge (or even fade away), including LowCode, Digital Twins and Digital Shadows, to address the increasing complexity of configurability.

Even in areas of scientific computation, it is being recognized that modeling and executable programming languages are not enough to compare and realize the full potential of scientific models. For progress to occur, it would be helpful if analysis techniques could be applied directly on the models to allow a scientist to understand the different encodings of scientific properties and thus predict the quality of scientific models with respect to reality. This is especially interesting when the quality of models has direct political, societal and industrial consequences, such as the case with climate models.

We are eager to see the outcomes of these efforts and whether we will be able to abstractly model systems and analyze their desired and emerging properties before physically creating a first prototype. Because software is an intrinsic part of all kinds of innovative systems, we can also expect a need for modeling the software part within these systems, which may give the UML and UML-related modeling languages a new push.

1 Content of this issue

1. **Expert Voice**
   - "Model-based intelligent user interface adaptation: Challenges and future directions" by Silvia Abrahao, Emilio Insfran, Arthur Sluyters, and Jean Vanderdonckt

2. **BPMDS 2019 Special Section**
   - Guest Editor: Jens Gulden

3. **Regular Papers**
   - "MUPPIT: A method for using proper patterns in model transformations" by Mahsa Panahandeh, Mohammad Hamdaqa, Bahman Zamani, and Abdelwahab Hamou-Lhadj
   - "Model-driven development platform selection: four industry case studies" by Siamak Farshidi, Slinger Jansen, and Sven Fortuin
   - "Spectra: A specification language for reactive systems" by Shahar Maoz and Jan Oliver Ringert
   - "A modeling methodology for collaborative evaluation of future automotive innovations" by Maher Fakih, Oliver Klemp, Stefan Puch, and Kim Grüttner
   - "Model-based cloud resource management with TOSCA and OCCI" by Stéphanie Challita, Fabian Korte, Johannes Erbel, Faiez Zalila, Jens Grabowski, and Philippe Merle
   - "Coupling solvers with model transformations to generate explorable model sets" by Théo Le Calvar, Fabien Chhel, Frédéric Jouault, and Frédéric Saubion
   - "Handling nonconforming individuals in search-based model-driven engineering: nine generic strategies for feature location in the modeling space of the meta-object facility" by Jaime Font, Lorena Arcega, Oystein Haugen, and Carlos Cetina
   - "Distributed model validation with Epsilon" by Sina Madani, Dimitris Kolovos, and Richard Freeman Paige
   - "Automated generation of consistent, diverse and structurally realistic graph models" by Oszkár Semeráth, Aren Babikian, Boqi Chen, Chuning Li, Kristóf Marussy, Gabor Szarnyas, Daniel Varro
   - "Controllable and decomposable multidirectional synchronizations" by Gabor Bergmann

Funding Open Access funding enabled and organized by Projekt DEAL.