

Theme Section on Modeling for Simulation

Istvan David^{1,2} · Philipp Zech³ · Valdemar Vicente Graciano Neto⁴ · Deniz Cetinkaya⁵

Received: date / Accepted: date

1 Overview

Simulation has long been a fundamental instrument for understanding, predicting, and evaluating the behavior of complex systems, particularly when analytical or purely numerical approaches are cumbersome or infeasible [7]. Its role has expanded considerably over the past two decades [1]. Simulators are no longer confined to design-time analysis, but increasingly participate in the operation of systems themselves, supported by developments such as digital twins, virtual and augmented reality, and increasingly sophisticated cyber-physical systems. As a result, modern simulators are expected not only to provide high-fidelity and insightful representations of system behavior [6], but also to integrate with heterogeneous engineering processes, tools, models, and runtime data [2].

Despite this growing importance, the engineering of simulators and simulation environments still often lacks the systematic methods available in other areas of software and systems engineering. This complicates the verification and validation of simulations, the reuse and evolution of simulation components, the management of simulation models, and the preservation of traceability between models, simulations, and the systems they represent. Model-driven engineering [3] offers a natural foundation for addressing these challenges. High-level models, domain-specific languages, automated transformations, models at runtime, and

model-based integration techniques can support the systematic construction, configuration, execution, and evolution of simulations while making their underlying assumptions and relationships explicit [4, 5]. At the same time, the interaction between model-driven engineering and simulation technology remains a relatively underexplored area, particularly as simulation becomes increasingly distributed, adaptive, and intertwined with runtime system operation.

This theme section on *Modeling for Simulation* invited researchers and practitioners to explore how modeling can contribute to more systematic, robust, reusable, and scalable simulation engineering. The scope encompasses model-driven engineering of simulator architectures and components, verification and validation, traceability, domain-specific languages, simulation-model management, models at runtime, model-driven co-simulation, and the integration of modeling and simulation toolchains, as well as emerging applications such as digital twins and AI-enabled simulation. The articles in this theme section illustrate the breadth of this intersection, ranging from the engineering and integration of simulation infrastructures to the use of executable and runtime models for supporting analysis, adaptation, and sustainability.

2 Selected articles

The theme section received 13 submissions, of which five were selected for publication after a careful review process.

In their article on *The Role of Standardization for Simulation in Model-Based Systems Engineering: A Survey Study Supplemented with Industrial Ex-*

¹ McMaster University, Canada

² McMaster Centre for Software Certification (McSCert), Canada

³ University of Innsbruck, Austria

⁴ Universidade Federal de Goiás, Brazil

⁵ Bournemouth University, UK

periences, Erik Rosenlund, Johan Cederbladh, Per Strandberg, and Daniel Krems investigate the role of standardization in integrating modeling and simulation into MBSE through a survey of 50 researchers and practitioners, supplemented with four industrial experience reports. Their findings highlight domain-dependent trade-offs between rigor and agility while identifying cross-domain opportunities for standardization, particularly in support of simulation credibility, traceability, interoperability, and model reuse.

In their article on *Bridging Conceptual Models and Numerical Domains: A Semantic Integration Approach for Federated Co-Simulation in MBSE*, Thomas C. Zimmermann, Johan Cederbladh, Erik Paul Konietzko, and Pascal Lünemann present FUSION, a semantic integration framework that connects conceptual SysML v2 models with numerical simulation models for federated co-simulation. The framework combines knowledge graphs, ontology-based mappings, the Asset Administration Shell, and the Functional Mock-up Interface standard to support the discovery, configuration, and semantic validation of simulation models, and is demonstrated on a fuel-cell co-simulation scenario.

In their article on *Circulise4LCE, A Model-Driven Sustainability Development Framework for Local Circular Economy Engineering and Simulation*, Yves Wautelet and Xavier Rouget introduce Circulise4LCE, a model-driven framework for engineering local circular economies by connecting an i*-based organizational pattern with BPMN process models, UML design models, and executable simulation. The framework is demonstrated on the Fanyatu reforestation initiative in the Congo Basin, showing how socio-technical stakeholder structures and circular resource flows can be carried from conceptual modeling into a simulation environment for analyzing and evaluating sustainability solutions.

In their article on *Modeling Structured Programming with Strudel: A Simulation Runtime for Programming Education Tools*, André Santos and Afonso Caniço present Strudel, an executable modeling language and simulation runtime that retains models of structured programs at runtime to support fine-grained observation of program execution. The approach is implemented as an embeddable virtual machine and underpins educational tools for tracing, white-box assessment, and automated questions about program behavior; its practical viability is demonstrated in a Java programming course with more than 400 students.

In their article on *A Model-Driven Framework for Supervised Reconfiguration of Satellite Schedulers*, Arianna Fedeli, Francesco Basciani, Daniele Masti, Ludovico Iovino, Paolo Serri, and Patrizio Pelliccione present a model-driven decision-support framework for adapting satellite scheduler configurations at runtime while retaining human supervision in safety-critical settings. The approach combines a Digital Twin of the scheduler with model-based specification and validation, telemetry-driven monitoring, simulation-assisted optimization, and generation of deployable configurations, and is evaluated through an industrial satellite scheduling scenario, an expert study, and quantitative validation of the model-driven toolchain.

Acknowledgments

We thank Editors-In-Chief Marsha Chechik, Benoit Combemale, Jeff Gray, and Bernhard Rumpe; as well as Martin Schindler from the SoSyM editorial office for their support with this theme issue. We are grateful to the reviewers for their invaluable contribution to the review of the manuscripts, and to the authors for choosing SoSyM to publish their insightful works.

References

1. Boschert S, Rosen R (2016) Digital twin—the simulation aspect. In: *Mechatronic futures*, Springer, pp 59–74
2. Liu X, David I (2025) AI simulation by digital twins: Systematic survey, reference framework, and mapping to a standardized architecture. *Software and Systems Modeling* DOI 10.1007/s10270-025-01306-0
3. Schmidt DC (2006) Model-driven engineering. *Computer-IEEE Computer Society*- 39(2):25
4. Syriani E, Vangheluwe H (2008) Programmed graph rewriting with time for simulation-based design. In: *Theory and Practice of Model Transformations*, Springer, pp 91–106
5. Syriani E, Vangheluwe H (2013) A modular timed graph transformation language for simulation-based design. *Software & Systems Modeling* 12(2):387–414, DOI 10.1007/s10270-011-0205-0
6. Zech P, David I (2026) Model-driven data contracts for digital twin services. In: *2026 ACM/IEEE 29th International Conference on Model Driven Engineering Languages and Systems (MODELS)*, ACM
7. Zeigler BP, Muzy A, Kofman E (2018) *Theory of modeling and simulation: discrete event & iterative system computational foundations*. Academic press