

# Conference Paper Title\*

\*Note: Sub-titles are not captured in Xplore and should not be used

1<sup>st</sup> Nassim Awabdy  
Fachbereich 5  
Aachen University of Applied Sciences  
Aachen, Germany  
nassim.awabdy@alumni.fh-aachen.de

**Abstract**—This document is a model and instructions for L<sup>A</sup>T<sub>E</sub>X. This and the IEEETran.cls file define the components of your paper [title, text, heads, etc.]. \*CRITICAL: Do Not Use Symbols, Special Characters, Footnotes, or Math in Paper Title or Abstract.

**Index Terms**—cyber-physical systems, mbse, sysml, modeling guidelines

## I. INTRODUCTION

Modern Cyber-Physical Systems (CPS) are technical systems that combine mechanical, electronic, and software subsystems with physical elements embedded in the real world. The development of CPSs is becoming increasingly complex and challenging, due to their interdisciplinary nature and the need to ensure seamless integration between their physical and computational components [2], [3].

Model-Based Systems Engineering (MBSE) is a methodology for the development and management such complex systems, that addresses issues arising from the complexity and interdisciplinary nature of CPS, and provides the agility required to adapt to changing requirements and technologies. MBSE incorporates a centralized system model as the primary source of information, throughout the system lifecycle [2]–[7], [13], [15]–[17].

SysML v1 has been widely adopted as the standard for modelling CPS and served as a key enabler MBSE. SysML v1 is a graphical, general purpose modelling language that is defined as an extension of the Unified Modeling Language (UML). Because it was built on top of UML, SysML v1 inherited several limitations from UML, that limited its expressiveness and usability for CPS modelling. However, it still provided a solid foundation for specifying and analyzing a system's behaviour, structure and requirements [3], [4], [7], [9], [11].

The release of SysML v2 represents the next generation of the Systems Modeling Language, designed as a comprehensive overhaul of SysML v1 that address its limitations and enhance the efficacy of MBSE practices. Unlike its predecessor, SysML v2 is built upon the Kernel Modeling Language (KerML), this approach ensures that SysML v2 inherits a formal semantic foundation, that is crucial for enhanced precision and automation in MBSE workflows. [3], [4], [7], [9], [11], [13].

Identify applicable funding agency here. If none, delete this.

While these advancements introduced by SysML v2 are promising, the abstract nature of the language still presents challenges for ensuring consistent modeling practices across diverse engineering teams. Model inconsistencies creates a high risk of redundant effort, potential modeling errors, and lack of reuse of system elements, preventing them from being aggregated into a coherent overall system model. Therefore, mechanisms for implementing and enforcing modeling guidelines must evolve to leverage the native formal capabilities of SysML v2, that enable effective model verification processes throughout the development lifecycle [2]–[4], [11], [13].

Within the context of model analysis, a distinct differentiation between model validation and model verification is necessary.

*Model validation* is a form of static analysis that ensures a model's structural and semantic correctness. By detecting inconsistencies ranging from syntactic errors to mismatched data types, validation guarantees the model's precision and coherence. This process as a critical quality control, ensuring the model is reliable before it undergoes transformation or dynamic verification [4], [16].

*Model verification*, conversely, is a dynamic analysis technique focused on ensuring that the system specified by the model behaves exactly as intended under various conditions, thereby confirming behavioral adherence to specifications [16].

This work investigates the difference between mechanisms for implementing modeling guidelines in SysML v1 and SysML v2. We analyze the implications these differences have on model verification processes, by looking into a couple of specific scenarios.

## II. THEORETICAL BACKGROUND

### A. Model-Based Systems Engineering (MBSE)

Model-Based Systems Engineering (MBSE) is defined by the International Council on Systems Engineering (INCOSE) as "The formalized application of modeling to support system requirements, design, analysis, verification, and validation activities beginning in the concept stage and continuing throughout development and later life cycle stages." [8]. Unlike traditional document-based approaches, where system's information is scattered across various files, MBSE is a methodology that centralizes system's information within a unified model, which serves as the primary source of information that

is managed throughout the system lifecycle [7]. This approach enhances collaboration among multidisciplinary teams, reduces errors from manual synchronization, and enables system architects to respond more quickly and effectively to numerous changes in requirements that occur during the development process [1], [6], [15].

### B. SysML v1 Foundations

SysML v1 is a graphical, general purpose modeling language that is widely recognized as the standard language for MBSE, serving as a foundational tool for specifying, analyzing, designing and verifying complex multidisciplinary systems [1], [4], [9]. SysML v1, adopted by the Object Management Group (OMG) in 2007, was essential in advancing MBSE practice by providing capabilities for formally capturing system requirements, structure, behaviour, and parametrics [7]. The language is defined as an extension of the Unified Modeling Language (UML), allowing it to adopt established modeling concepts [4].

SysML v1 organizes its modeling constructs into four main categories:

- **Structure:** This aspect is modeled using Block Definition Diagrams (BDDs) and Internal Block Diagrams (IBDs). BDDs are used to define components, interfaces, and relationships at black-box level, while IBDs offer a white-box perspective by outlining the internal structure of a single block [9].
- **Behaviour:** SysML v1 provides several behavioral diagrams, including State Machine Diagrams, Sequence Diagrams, and Activity Diagrams [9], [15]. These diagrams capture distinct uses for modeling system behaviour.
- **Requirements:** System requirements are specified in Requirement Diagrams, that largely rely on natural language representation, although they can be linked logically to other model elements [9], [11].
- **Parametrics:** Parametric Diagrams define mathematical constraints and equations between system elements. They support preliminary calculations and analysis within the model, though they often require external tools for complex evaluations [9].

SysML v1 allows the restriction of modeling practices through UML profiling mechanism, enabling the construction of specialized extensions. The central element for customization is the **stereotype**, which functions as a distinct metaclass within UML [1]. Stereotypes enable the customization of existing metaclasses by associating them with specific properties and constraints, thereby tailoring the modeling language to meet domain-specific requirements [1], [14].

The **Object Constraint Language (OCL)** function as a specialized textual specifications language that enables formally defining rules, constraints and queries on models especially those created using UML [13]. It was developed to overcome the limitations of graphical modeling by allowing precise specification of rules that cannot be easily represented visually [1].

### C. SysML v2 Foundations

SysML v2 represents a major evolution over its predecessor, having been engineered independently from UML to overcome the limitations inherited from it [4], [7]. It aims to enhance MBSE adoption and effectiveness by focusing on improving precision, expressiveness, consistency, usability, interoperability, and extensibility. [7] The foundation of SysML v2 is built upon a new general-purpose modeling language called the **Kernel Modeling Language (KerML)** [11].

The mechanism of KerML is built upon a hierarchical, three-layered architecture, successively progressing from general to specific constructs [12].

- The *Root Layer* establishes the essential syntactic scaffolding for constructing models. The main focus of this layer is to define organizational constructs, such as *Elements*, *Namespaces*, and *Relationships*, leaving out model-level semantic interpretation relative to the modeled system [11], [12].
- The *Core Layer* introduces the language's semantic foundation based on classification, defined in first-order logic axioms. The central primitive is *Type*, which is divided into *Classifiers*, *Features* and key relationships between them necessary for organizing classification hierarchies [11], [12].
- The *Kernel Layer* finalizes the language specification by adding specialized constructs used in common modeling applications, such as *Data Types*, *Classes*, *Structures*, and *Behaviours* [11], [12].

KerML achieves its consistent semantics through formal mathematical logic and library-based ontological modeling that maintain a precise interpretation of complex models. Since the semantics in the *Core Layer* are defined using first-order logic, a consistent basis for mathematical reasoning about models is established [11], [12]. For comprehensive concepts introduced in the *Kernel Layer*, KerML extends its semantics through the reuse of elements found in the **Kernel Semantic Library**. This library is itself expressed in KerML, meaning that all concepts in the language are ultimately grounded in the same formal semantic framework [11], [12].

[4], [11], [13]

### III. RESULTS

- A. *SysML v1 Implementation of Modeling Guidelines*
- B. *SysML v2 Implementation of Modeling Guidelines*
- C. *Comparative Analysis*

### IV. DISCUSSION AND IMPLICATIONS

### V. CONCLUSION AND OUTLOOK

### ACKNOWLEDGMENT

### REFERENCES

- [1] L. Beers, H. Nabizada, M. Weigand, F. Gehlhoff, and A. Fay, “A sysml profile for the standardized description of processes during system development,” in 2024 IEEE International Systems Conference (SysCon). IEEE, 2024, pp. 1–8.
- [2] S. Bergemann, “Challenges in multi-view model consistency management for systems engineering,” in Modellierung 2022 Satellite Events. Bonn: Gesellschaft f”ur Informatik e.V., 2022, pp. 77–89.
- [3] K. Boelsen, M. May, G. Jacobs, and et al., “Sysml v2 based modelling guidelines for mechanical system elements,” *Forsch Ingenieurwes*, vol. 89, no. 60, 2025.
- [4] E. Cibrián, J. Olivert-Iserte, C. Díez-Fenoy, R. Mendieta, J. Llorens, and J. M. Álvarez Rodríguez, “Ensuring semantic consistency in sysml v2 models through metamodel-driven validation,” *IEEE Access*, vol. 13, pp. 121 444–121 457, 2025.
- [5] S. Dehn, G. Jacobs, P. Höck, and G. Höpfner, “Enhancing model-based development with formalized requirements: integrating temporal logic and sysml v2 for comprehensive state and transition modeling,” *Forschung im Ingenieurwesen*, vol. 89, no. 1, p. 53, 2025.
- [6] P. Dukić, “Generating sysmlv2 models from structured natural language,” 2025.
- [7] S. Friedenthal, “Future directions for mbse with sysml v2.” in MODEL-SWARD, 2023, pp. 5–9.
- [8] D. D. Walden, G. J. Roedler, K. J. Forsberg, R. D. Hamelin, and T. M. Shortell, Eds., *INCOSE Systems Engineering Handbook: A Guide for System Life Cycle Processes and Activities*, 4th ed. Hoboken, NJ, USA: John Wiley & Sons, 2015.
- [9] N. Jansen, J. Pfeiffer, B. Rumpe, D. Schmalzing, and A. Wortmann, “The language of sysml v2 under the magnifying glass.” *J. Object Technol.*, vol. 21, no. 3, pp. 3–1, 2022.
- [10] H. Kausch, M. Pfeiffer, D. Raco, B. Rumpe, and A. Schweiger, “Model-driven development for functional correctness of avionics systems: a verification framework for sysml specifications,” *CEAS Aeronautical Journal*, vol. 16, no. 1, pp. 33–48, 2025.
- [11] V. Molnár, B. Graics, A. Vörös, S. Tonetta, L. Cristoforetti, G. Kimberly, P. Dyer, K. Giamarco, M. Koethe, J. Hester et al., “Towards the formal verification of sysml v2 models,” in Proceedings of the ACM/IEEE 27th International Conference on Model Driven Engineering Languages and Systems, 2024, pp. 1086–1095.
- [12] Object Management Group, “Kernel Modeling Language (KerML), Version 1.0,” Object Management Group, Specification formal/25-09-01, Sep. 2025. [Online]. Available: <https://www.omg.org/spec/KerML/1.0/PDF>
- [13] A. Ratzke, J. Koch, and C. Grimm, “Modeling and analysis of system models with constraints in sysml v2,” in 2025 20th Annual System of Systems Engineering Conference (SoSE), 2025, pp. 1–6.
- [14] S. Raedler, J. Mangler, and S. Rinderle-Ma, “Model-driven engineering method to support the formalization of machine learning using sysml,” arXiv preprint arXiv:2307.04495, 2023.
- [15] U. Yildirim, F. Campean, A. Korsunovs, and A. Doikin, “Flow heuristics for functional modelling in model-based systems engineering,” *Proceedings of the Design Society*, vol. 3, pp. 1895–1904, 2023.
- [16] Zavada, G. Kulcsár, V. Molnár, and Horváth, “Towards a configurable verification and validation framework for critical cyber-physical systems,” in 2025 IEEE 8th International Conference on Industrial Cyber-Physical Systems (ICPS), 2025, pp. 1–6.
- [17] Z. Li, F. Faheem, and S. Husung, “Collaborative model-based systems engineering using dataspaces and sysml v2,” *Systems*, vol. 12, no. 1, 2024. [Online]. Available: <https://www.mdpi.com/2079-8954/12/1/18>