systems

MDPI

Article

MBSE 2.0: Toward More Integrated, Comprehensive, and Intelligent MBSE

Lin Zhang [1,2,3], Zhen Chen [1,2,3,*], Yuanjun Laili [1,2,3], Lei Ren [1,2,3], M. Jamal Deen [4], Wentong Cai [5], Yuteng Zhang [1,2,3], Yuqing Zeng [1,2,3] and Pengfei Gu [1,2,3]

1 Hangzhou International Innovation Institute, Beihang University, 166 Shuanghongqiao Street, Pingyao Town, Yuhang District, Hangzhou 311115, China; zhanglin@buaa.edu.cn (L.Z.)
2 School of Automation Science and Electrical Engineering, Beihang University, Xueyuan Road No. 37, Haidian, Beijing 100191, China
3 State Key Laboratory of Intelligent Manufacturing Systems Technology, Yongding Road No. 51, Haidian, Beijing 100854, China
4 AI Atlas Inc., Hamiton, ON L8S 4K1, Canada
5 School of Computer Science and Engineering, Nanyang Technological University, Singapore 639798, Singapore
* Correspondence: czhen@buaa.edu.cn

Abstract

Model-Based Systems Engineering (MBSE) has gained significant attention from both industry and academia as an effective approach to managing product complexity. Despite its progress, current MBSE concepts, tools, languages, and methodologies face notable challenges in industrial applications, particularly in addressing design variability, ensuring model consistency, and enhancing operational efficiency. Based on the authors' industry observations and literature analysis, this paper identifies the primary limitations of traditional MBSE, and introduces MBSE 2.0, a next-generation evolution characterized by comprehensive, integrated, and intelligent features. Key enabling technologies, such as model governance, integrated design methods, and AI-enhanced system design, are explored in detail. Additionally, several preliminary explorations were introduced under the guidance of the MBSE 2.0 philosophy. This study introduces the MBSE 2.0 concept to stimulate discussion and guide future efforts in academia and industry, emphasizing key advancements and highlighting several key and pressing perspectives to alleviate current limitations in industrial practice.

Keywords: MBSE; system design; smart manufacturing; cyber-physical systems

check for updates

Academic Editor: Timothy Ferris

Received: 6 June 2025
Revised: 2 July 2025
Accepted: 9 July 2025
Published: 15 July 2025

Citation: Zhang, L.; Chen, Z.; Ren, L.; Yuanjun, L.; Deen, M.J.; Cai, W.; Zhang, Y.; Zeng, Y.; Gu, P. MBSE 2.0: Toward More Integrated, Comprehensive, and Intelligent MBSE. *Systems* **2025**, *13*, 584. https://doi.org/10.3390/systems13070584

Copyright: © 2025 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

1. Introduction

Model-Based Systems Engineering (MBSE) is an approach that supports system requirements, design, analysis, verification, and validation through structured and standardized models [1]. MBSE enhances multidisciplinary integration and consistent understanding among developers [2].

The increasing complexity of modern systems, driven by rapid technological advancements and rising expectations for performance, has made MBSE increasingly indispensable. Industries such as aerospace, automotive, and heavy equipment widely adopt MBSE to develop safer, smarter, and more cost-effective products with shorter development cycles [3,4]. Consequently, global technology leaders such as ANSYS, HCLTech, Siemens, and MathWorks have significantly expanded their investments in MBSE, substantially accelerating its development and market growth [5]. For instance, Airbus employs MBSE

across the entire lifecycle of the A350 aircraft [6], NASA actively promotes its applications in aerospace projects [7], and COMAC has recently integrated MBSE methodologies into the development of its latest aircraft models [8]. According to the MBSE Tools Market Report by DataIntelo (https://dataintelo.com/report/global-model-based-systems-engineering-mbse-tools-market (accessed on 1 November 2024), the global MBSE solutions market was valued at approximately USD 2.5 billion in 2023 and is expected to grow at a compound annual growth rate (CAGR) of 13.2% over the next decade, underscoring the increasing reliance on MBSE technologies across industries. Despite widespread adoption and market growth, current MBSE practices still face challenges in integrating lifecycle stages, achieving automation and intelligence, and supporting comprehensive system modeling. These gaps highlight the need for further methodological innovation.

The foundational elements of MBSE (language, methodology, and tools) have evolved significantly over the past decades [9,10]. Current modeling languages, particularly semi-formal languages such as SysML, UML, Simulink, and Modelica, dominate MBSE applications but face challenges in supporting integrated simulation and verification. A review by [11] summarizes the usage of existing MBSE languages and tools, as shown in Figure 1. Although methodologies and tools have matured, significant gaps remain regarding interoperability, full-lifecycle coverage, and intelligent automation capabilities [12–14]. The exponential growth of system complexity, digitalization, and unstructured data challenges in the era of big data and artificial intelligence demands a fundamental evolution beyond traditional MBSE methods.

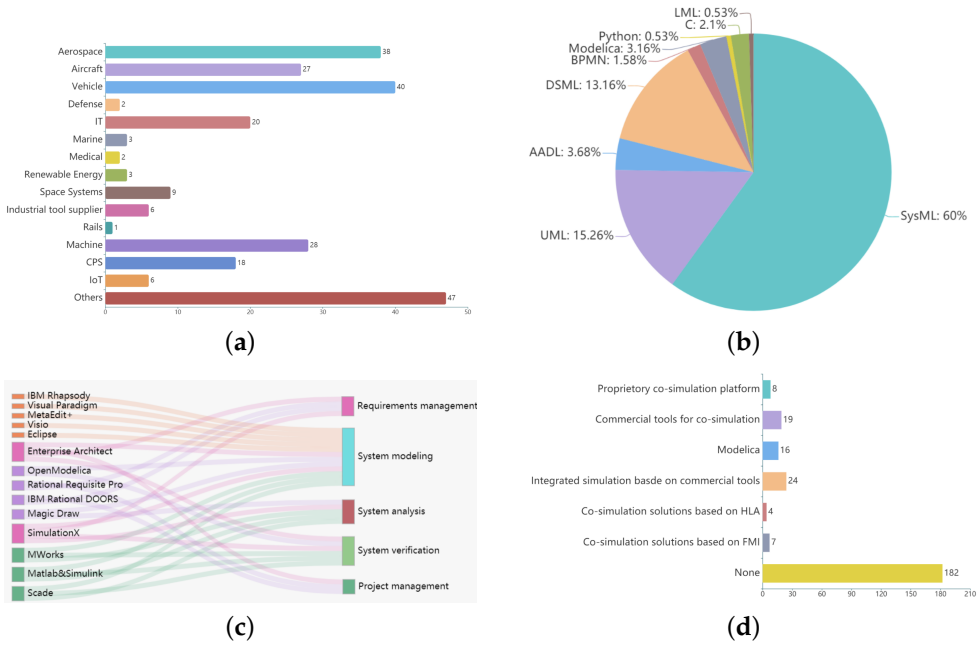

Figure 1. The status of MBSE applications, languages, and tools as presented in the literature [11]. (**a**) Application areas of MBSE toolchains. (**b**) MBSE modeling language usage. (**c**) MBSE tool classification. (**d**) Integrated simulation methods. Note that Capella, a prominent MBSE tool based on the Arcadia methodology, is not included in this figure. It primarily supports system modeling and system analysis.

Over recent years, MBSE has made significant strides, with the development of new languages, tools, and methodologies. Notable advancements include the Magic Grid method, proposed by No Magic (now part of Dassault Systemes) in 2017, which integrates OOSEM principles and SysML to facilitate comprehensive product development [15]. Magic Grid, implemented using the MagicDraw tool [16], provides a matrix-view framework that enhances the modeling and simulation of system requirements and architecture.

In 2020, Lu et al. introduced the Karma language, which formalizes models using the Graph, Object, Point, Property, Role, and Relationship (GOPPRR) approach [17]. This method bridges gaps between different modeling tools and languages. Similarly, Zhang et al.'s 2021 proposal of the X language integrates elements from SysML, Modelica, DEVS, and BDI specifications, aiming to unify system architecture modeling with physical characteristic modeling at the semantic level, enabling direct simulation and verification [18]. Recent developments in systems engineering standards have significantly shaped model-based practices. ISO/IEC/IEEE 15288:2023 [19] provides a standardized framework of 30 life-cycle processes and shared terminology spanning concept, development, operation, support, and retirement of human-made systems. ISO/IEC/IEEE 24641:2023 [20] complements this by defining a reference model for Model-Based Systems and Software Engineering (MBSSE), detailing processes, tool methods, and interoperability requirements for model-centric engineering across life-cycle stages. In parallel, SysML 2.0 [21], released by OMG in 2024, marks a step-change in system modeling languages, offering richer semantic expressiveness and improved interoperability across engineering tools. However, these standards still largely rely on structured data and human-intensive modeling workflows. They do not fully address emerging challenges related to unstructured data integration, continuous automated verification and validation, intelligent generative modeling, or deep lifecycle integration from initial design to disposal. These gaps significantly limit the agility and intelligence needed for next-generation complex system designs.

Despite significant advancements, current MBSE methods still face critical challenges: aligning dynamic requirements with rigid architectures, tool integration complexities, limited lifecycle application, low automation and intelligence, and poor interoperability across models and tools:

1. **Aligning Design with Requirements**: Timely alignment between design and requirements remains a challenge. In practice, fluctuating user requirements complicate the clear definition of system objectives. Systems are often prematurely architected, hindering optimization at the architectural level. Furthermore, many system architectures are not sufficiently adaptable to changing requirements [22].
2. **Tool Integration and Simulation Challenges**: Integrating system modeling and simulation tools remains a significant hurdle. Different tools have disparate core capabilities, making efficient system design and validation difficult. For instance, SysML-based tools excel in modeling but lack simulation functionality, while simulation tools like Simulink and Modelica are limited in system modeling capabilities [12]. Although some integration methods exist, as shown in Figure 1d, they do not fully address issues such as model modification, feedback, and traceability. Also, the wide variety of tools, inconsistent modeling and simulation guidelines, and unclear standards make tool interface construction and debugging very challenging.
3. **Limited application in product lifecycle.** This means that most MBSE implementations are confined to certain phases of the system design process. While MBSE emphasizes a model-based approach throughout the entire product lifecycle, it is often not applied beyond the early stages of design, leaving gaps in later lifecycle phases [23]. This lack of full-lifecycle integration limits the ability to make informed decisions throughout product development. For example, engineers lack guidance on using MBSE for activities across other phases of the product's full lifecycle, while managers face difficulties in making effective decisions from a full lifecycle perspective to drive MBSE-guided product development.
4. **Low Automation and Intelligence in MBSE Toolchains**: While existing MBSE tools can handle basic modeling and simulation needs, their automation and intelligence levels are still inadequate [24]. Engineers often spend significant time operating the

tools, which detracts from the focus on actual system design [13]. These tools do not yet provide the level of operational efficiency needed for highly complex systems.

5. **Lack of Unified or Modular Model Representations**: The diversity of standards across industries and tools hinders model reuse and integration [24]. Even tools using the same modeling language often face interoperability issues, requiring conversion plugins for data exchange. This complexity undermines the efficiency of model and data integration in the development of complex products.

These challenges expose fundamental limitations of traditional MBSE methods, highlighting the urgent need for a more intelligent, integrated, and comprehensive framework—MBSE 2.0.

To overcome these limitations, researchers have explored new perspectives and trends in MBSE development. MBSE is increasingly integrating with emerging technologies such as digital twins and digital threads to extend its applications across the entire production lifecycle [25,26]. This integration aims to create a seamless model-driven process from requirements analysis to system architecture design, simulation, and optimization, ensuring consistency and reducing information loss between different stages [27]. Additionally, standards and protocols are being actively developed to unify various design phases and tools, promoting greater interoperability and unification in system design practices [28–30]. More critically, artificial intelligence and advanced simulation techniques have become key enablers of the intelligent transformation of MBSE [31–33]. AI can significantly enhance the automation of model creation, accelerate real-time verification and validation processes, dynamically adapt to changing requirements, and provide valuable insights during system development. We term this evolved, intelligent, deeply integrated methodology "MBSE 2.0", clearly distinguishing it from its predecessors.

MBSE 2.0 introduces a new paradigm in systems engineering characterized by deep integration across lifecycle stages, advanced intelligent automation, and comprehensive model-driven management. This paper clearly defines MBSE 2.0, outlines its core features and benefits, and systematically describes the enabling technologies and practical application scenarios. Specifically, Section 2 introduces MBSE 2.0 concepts and unique characteristics. Section 3 details key enabling technologies underpinning MBSE 2.0, and Section 4 demonstrates real-world scenarios leveraging MBSE 2.0 to effectively address the outlined challenges.

2. Preliminaries

In this section, we review the main existing MBSE methods, further analyze the shortcomings of existing methods, and explain the motivation and goals of this paper.

2.1. State of the Art

The International Council on Systems Engineering (INCOSE) introduced the concept of Model-Based Systems Engineering (MBSE) in its Vision 2020 report [34], defining it as a formalized modeling approach spanning the entire system lifecycle. MBSE addresses the limitations of document-centric engineering by replacing informal textual descriptions with structured, standardized models [35,36], thus enhancing consistency, communication, and collaboration across stakeholders. To support its adoption, INCOSE and the Object Management Group (OMG) jointly developed SysML based on UML [37], which has since become the foundational modeling language for mainstream MBSE methodologies. SysML v2 was proposed recently, and it introduces a new metamodel and textual notation to improve expressiveness, precision, and tool interoperability. It also enhances support for model-based analysis, reuse, and integration with digital engineering ecosystems.

A variety of MBSE methodologies have been proposed to guide the implementation of model-based engineering practices [38]. Harmony-SE, proposed by IBM, follows the classic V-model, encompassing requirement analysis, functional analysis, and design synthesis, supported throughout by SysML [39,40]. OOSEM, developed by Lockheed Martin and INCOSE, integrates object-oriented principles with model-based design, using a top-down modeling approach facilitated by SysML [41,42]. RUP SE, derived from the Rational Unified Process, adopts parallel design and iterative development to address the complexities of system engineering [43–45]. The Vitech MBSE approach employs the onion model, emphasizing holistic top-level design followed by layered refinement to reduce risk and ensure structural integrity [46,47]. State Analysis (SA), proposed by JPL, focuses on control systems through explicit state modeling, aiming to bridge the gap between system requirements and software implementation [48,49]. The Object-Process Methodology (OPM), developed by Dov Dori, unifies object and process perspectives using both graphical (OPD) and textual (OPL) representations [50–52]. MagicGrid, developed by No Magic based on OOSEM, adopts a matrix-based framework dividing development into requirement, architecture, and implementation domains, all supported by SysML [53,54]. Arcadia, developed by Thales, provides an architecture-centric MBSE framework supported by the Capella tool. It facilitates multi-view modeling of system, logical, and physical architectures, ensuring traceability and consistency throughout the engineering lifecycle [55]. In addition, domain-specific MBSE approaches such as FAS4M, SPES, and Motego have been proposed [56,57]. FAS4M targets mechanical system components using MechML for multi-view modeling, though lacking in physical modeling and validation [58,59]. The SPES methodology, based on the SPESML language, manages model granularity for embedded systems [56]. Motego, following an RFLP (Requirement–Function–Logic–Product) structure, supports electromechanical product development and incorporates physical modeling capabilities [57,60–62]. In summary, except for Motego, most existing MBSE methodologies focus primarily on requirements, functions, and logical models, with limited support for multidisciplinary physical modeling and comprehensive verification. While Motego addresses physical aspects, it relies on heterogeneous tools and languages across layers, increasing the complexity of maintaining model consistency [57].

The selection and integration of modeling tools play a critical role in the promotion of MBSE practices. Cameo Systems Modeler (formerly MagicDraw) stands out for its comprehensive SysML support, built-in Model Simulation Toolkit, and seamless integration with multidisciplinary optimization (MDO) platforms (e.g., ModelCenter, Isight) and physical simulation tools (e.g., Simulink, Modelica), excelling in dynamic simulation and multi-objective optimization. It also enables collaborative modeling and version control through Teamwork Cloud [63]. In contrast, Enterprise Architect by Sparx Systems supports multiple modeling languages (SysML, UML), offering high cost-effectiveness and cross-domain applicability. While it lacks deep simulation capabilities, it is widely adopted in SMEs and academic contexts. IBM Rhapsody targets embedded and safety-critical systems, emphasizing real-time simulation and model-driven development, though its ecosystem and cross-platform support are more limited [64]. Capella, built upon the Arcadia methodology, focuses on system architecture design and early-stage solution assessment. Although less suited for deep simulation, it offers strong support for architecture analysis and integration with traditional engineering workflows [65]. Collectively, these tools complement each other across different application domains, forming a closed-loop workflow from system modeling to performance evaluation.

2.2. Problem Statement and Motivation

The existing modeling tools still exhibit significant limitations in terms of integration. On the one hand, discrepancies in data formats and model representations across tools often require manual intervention or middleware during cross-platform collaboration and data exchange, thereby reducing the overall level of automation. On the other hand, although some tools support integration with MDO platforms and simulation software, they generally lack robust capabilities in real-time data synchronization, bidirectional feedback, and holistic system-level optimization. As a result, it remains difficult to achieve truly seamless modeling and simulation verification in complex system design, which constrains design efficiency.

Overall, current MBSE approaches largely focus on sequential or parallel tool usage, leading to fragmented and complex system-level workflows. These limitations present significant challenges for achieving high agility, efficiency, and rapid iterative development—manifested in the five key challenges identified in the Introduction. To address these issues, we introduce the concept of MBSE 2.0, which aims to strengthen and evolve the foundational MBSE paradigm by incorporating recent advances and addressing the shortcomings of existing toolchains.

The motivation for this work stems from the urgent need to stimulate discussion and raise awareness about the future trajectory of MBSE, particularly in the context of rapid technological revolutions. As MBSE continues to emerge as the dominant methodology for complex system development, it becomes increasingly critical to identify and define the directions necessary for its continued advancement. It is important to emphasize that this paper primarily aims to introduce and advocate the conceptual framework and theoretical foundations of MBSE 2.0. Rather than providing a detailed, prescriptive implementation roadmap or exhaustive technical procedures, our objective is to highlight critical issues and clarify key directions for evolving MBSE methodologies to meet emerging industry challenges. Specifically, our contributions are fourfold:

(1) Identifying and analyzing the core challenges currently facing traditional MBSE practices.
(2) Clearly articulating the conceptual distinctions introduced by MBSE 2.0 compared with conventional MBSE.
(3) Systematically summarizing and emphasizing the enabling technologies that underpin this new paradigm.
(4) Provides some practical examples guided by the core ideas of MBSE 2.0.

We hope this foundational exploration serves as a catalyst for further research, discussion, and development efforts in both academia and industry, paving the way toward realizing the full potential of MBSE 2.0.

3. Concept and Characteristics of MBSE 2.0

In this work, we define MBSE 2.0 as the next-generation evolution of model-based systems engineering that deeply integrates traditional model-based methodologies with advanced artificial intelligence, big data analytics, digital engineering, and enhanced simulation technologies. Unlike traditional MBSE, which primarily focuses on structured and standardized models for early-phase system design, MBSE 2.0 addresses the demands of complex, dynamically evolving systems by incorporating intelligent, adaptive modeling techniques capable of handling both structured and unstructured data across the entire system lifecycle. Thus, MBSE 2.0 aims to provide intelligent, integrated, and comprehensive lifecycle solutions.

As an evolution of traditional MBSE methodologies, MBSE 2.0 emphasizes three critical dimensions: intelligence, integration, and comprehensiveness. In particular, MBSE 2.0

focuses on enhancing the capability to efficiently construct adaptable, verifiable, and unified system models through integrated and intelligent modeling approaches. Such models are expected to better accommodate rapid changes in requirements and support iterative engineering practices. Additionally, MBSE 2.0 seeks to extend model-driven methods more comprehensively across all lifecycle stages. This extension is facilitated by emerging infrastructures like the Model Bus, which aim to improve the connectivity and management of knowledge, data, and models throughout the lifecycle, from initial design through to final disposal. By targeting existing limitations in adaptability, automation, and lifecycle integration, MBSE 2.0 aspires to offer improved coherence and effectiveness in managing increasingly complex systems engineering challenges.

3.1. Deep Integration of System Modeling, Simulation, and Optimization

Under the MBSE 2.0 framework, the deep integration of system modeling, simulation, and optimization is regarded as a fundamental advancement beyond conventional practices. Traditional MBSE approaches are often distributed across multiple stages and tools, as illustrated in Figure 2, resulting in fragmented workflows where system architecture, physical modeling, and performance optimization are handled separately, with limited real-time coordination and feedback. In the vast and complex product development process, product design iterations make such multi-stage integration-based solutions insufficient in terms of model consistency and traceability.

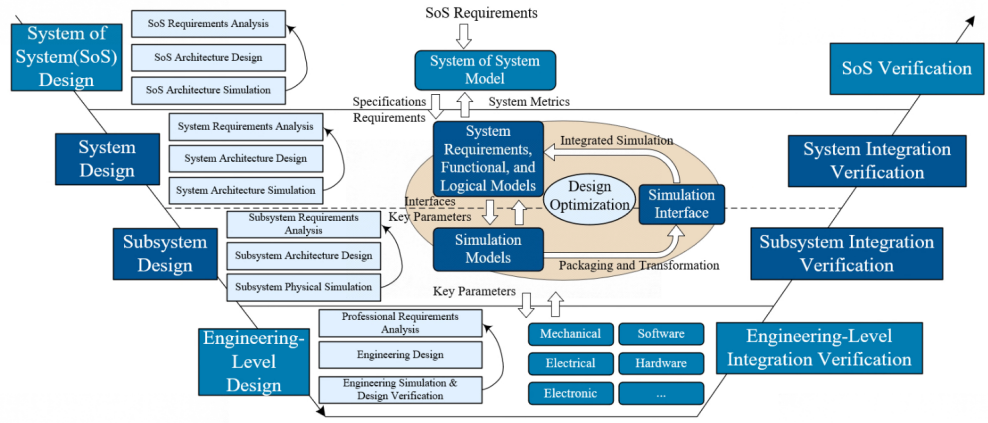

Figure 2. An overview of the traditional hierarchical and phased system design process, encompassing the full design flow from System-of-Systems (SoS) modeling down to engineering-level design and verification. A central model-based integration hub facilitates coordination between system requirements, functional and logical models, and simulation models, enabling limited feedback through optimization and interface packaging. However, modeling, simulation, and optimization remain loosely coupled, with verification activities conducted separately at each integration level, thereby highlighting the fragmented nature of traditional design workflows.

In contrast, MBSE 2.0 emphasizes a unified modeling paradigm in which system requirements, functional-logical structures, physical characteristics, and optimization processes are coherently represented and interconnected throughout the entire design lifecycle. As illustrated in Figure 3, within MBSE 2.0, the creation, updating, and maintenance of system design models, including system architecture, system physical characteristics, and optimization models, follow standardized and often uniform rules and formats. Such a framework enables cross-level model collaboration, where system optimization is not restricted to a single layer but is considered comprehensively across the entire system design, ultimately enhancing overall system performance and efficiency. Compared with current standards such as ISO/IEC/IEEE 15288:2023 [19] and SysML 2.0 [30], MBSE 2.0 specifically

extends their principles by embedding simulation and optimization routines directly within the modeling lifecycle, facilitating real-time design refinement and verification.

This proposed paradigm has been partially validated through previous works by the authors, such as the development of the X Language [18] and the SES-X methodology [66], which support unified modeling of system architecture and physical simulation. These efforts demonstrate one concrete implementation of the proposed integration concept, enabling seamless transitions between high-level architectural abstractions and executable system-level simulation verification. Nevertheless, these implementations primarily address the coupling between modeling and simulation. Further research is required to incorporate system-level optimization models into this integrated framework, enabling real-time architectural scheme evaluation and parameter optimization within the modeling lifecycle.

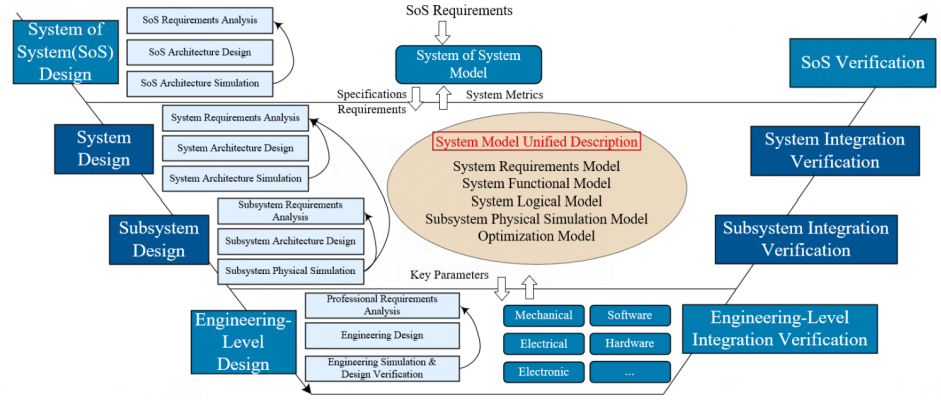

Figure 3. A schematic illustrating the deep integration of system modeling, simulation, and optimization, where architecture, simulation, and optimization models are unified within a continuous, semantically consistent modeling framework. Unlike Figure 2, which reflects the traditional MBSE workflow based on tool-level interfaces or loose coupling between models, this figure emphasizes a tightly integrated modeling process in which system requirements, logical and physical structures, simulation behaviors, and optimization objectives are coherently described and iteratively refined within a single modeling environment. This unified process enables more efficient design iterations, real-time feedback, and seamless transitions between abstraction layers.

3.2. Intelligent Analysis, Modeling, and Evaluation

Recent studies have explored the potential of artificial intelligence (AI) technologies—particularly natural language processing (NLP) and large language models (LLMs)—to support various aspects of model-based systems engineering (MBSE). For example, Longshore et al. [67] investigated the ability of LLMs such as ChatGPT-4 to generate and modify SysML v2 models through textual prompts, demonstrating feasibility on simple structural models. Similarly, Patel et al. [68] proposed a conceptual NLP pipeline to extract requirements and SysML entities from unstructured documents, aiming to lower the entry barrier to MBSE adoption. Other efforts have focused on the integration of AI across MBSE, MDAO (Multidisciplinary Design Analysis and Optimization), and digital twin environments, aiming to improve traceability, surrogate modeling, and operational prediction [69]. Additionally, refs. [70,71] outlined promising future directions, envisioning AI as a catalyst for intelligent systems engineering.

However, despite these encouraging developments, current research remains largely fragmented and tool-centric. Most implementations target isolated tasks—such as requirement extraction or model synthesis. Furthermore, limitations such as dependency on prompt engineering, lack of semantic validation, and insufficient integration across system domains hinder the realization of truly intelligent, adaptive MBSE workflows.

MBSE 2.0 framework calls for a comprehensive and systematic transformation of systems engineering practices through the deep integration of AI. Within this framework, AI technologies are not simply used as assistants for isolated tasks but are embedded across the entire lifecycle to enhance intelligence in analysis, modeling, and evaluation processes. As illustrated in Figure 4, this shift towards intelligent systems engineering aims to automate routine modeling operations, dynamically maintain model coherence, and provide real-time, data-driven feedback throughout development. A unified architectural mechanism to coordinate AI-driven consistency and reasoning across lifecycle stages is strongly needed. From the perspective of unified semantics, the unified system model descriptions presented in Section 2.1 are more conducive to facilitating intelligent generative system design.

Specifically, AI-driven generative modeling techniques in MBSE 2.0 can significantly reduce the manual workload associated with creating and maintaining system models. Technologies such as NLP and LLMs enable efficient extraction and structuring of requirements directly from various documentation sources, allowing engineers to focus on higher-level tasks like consistency checking and functional analysis. Beyond initial model generation, MBSE 2.0 emphasizes intelligent linking and mapping among requirements, behaviors, structures, and constraints using machine learning and model-reasoning techniques, ensuring continuous synchronization even under frequent design changes.

Furthermore, AI-enabled evaluation mechanisms, such as anomaly detection, surrogate-based verification, and intelligent simulation agents, can provide early and precise feedback, reduce costly rework, and accelerate design convergence. By integrating these capabilities across conceptual design, detailed modeling, and system validation stages, MBSE 2.0 envisions a cohesive, adaptive engineering paradigm that addresses both complexity and dynamism in modern systems.

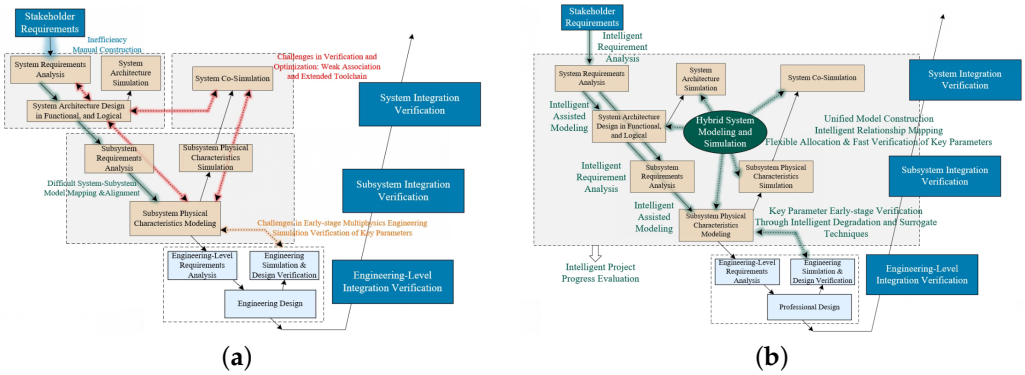

Figure 4. Comparison of the traditional system design pattern and the intelligent system design pattern. (**a**) A schematic of traditional phased manual modeling and verification. The system design process relies on labor-intensive analysis and complex tool operations. The stages of modeling, simulation, verification, and validation are loosely connected, with weak interdependencies and an extended toolchain. Mapping model relationships and maintaining consistency are challenging, hindering efficient system modeling and analysis. Early engineering-level simulation of key design parameters remains difficult. (**b**) A schematic of an intelligent system design empowered by artificial intelligence and integrated frameworks. This unified system modeling and simulation framework shortens the toolchain across the modeling, simulation, verification, and validation stages, leveraging AI, particularly generative AI, to enable intelligent association and validation. Key design parameters are validated at early engineering stages through intelligent agents or reduced-order modeling to achieve engineering-level simulation.

3.3. Rapid Construction of a Development Iteration Loop

In the system development process, MBSE 2.0 advocates the rapid construction of an iterative model-driven development loop, aiming to balance the inherent complexity of

modern systems with a streamlined and focused development core. By promoting the deep integration of system modeling, simulation, and optimization activities, and leveraging emerging intelligent technologies, MBSE 2.0 seeks to establish an efficient end-to-end development thread, from evolving system requirements through digital prototypes to physical prototypes.

This approach encourages the consolidation of critical development activities into a continuous, model-driven workflow, with the goal of mitigating design errors, reducing information loss across stages, and maintaining a clear focus on core system functionalities amidst increasing system complexity. MBSE 2.0 further promotes the adoption of agile engineering concepts, facilitating rapid design–prototype–test iteration cycles. Through such mechanisms, engineers are expected to iteratively refine system designs based on early-stage feedback, thereby accelerating project timelines while enhancing the adaptability and maturity of the evolving system.

By advocating rapid prototyping of key subsystems and emphasizing overall architecture optimization, MBSE 2.0 aims to support dynamic responses to shifting requirements while preserving system consistency. This balance between complexity management and development focus is anticipated to help reduce downstream modification and maintenance costs, improve engineering responsiveness, and strengthen the continuous alignment between system development efforts and project objectives.

3.4. Extension of Model-Driven Approaches to the Product Lifecycle

In MBSE 2.0, the model-based methodology is no longer confined to the system design process alone. Instead, it advocates extending model-driven approaches throughout all phases of the system lifecycle, aiming to enhance lifecycle continuity and data consistency. A key enabler of this extension is the concept of the Model Bus, as illustrated in Figure 5.

The Model Bus refers to a proposed technical architecture that manages and integrates heterogeneous system models generated at different lifecycle stages. It is intended to provide standardized interfaces, common data exchange formats, and unified access protocols, thereby facilitating seamless interaction, synchronization, and sharing of models across traditionally siloed development, production, and operational phases. Compared with conventional approaches such as Product Lifecycle Management (PLM) systems [72] or Digital Thread architectures [73], which primarily focus on data traceability and connectivity, the Model Bus emphasizes dynamic, model-centered interoperability. It advocates maintaining not just the traceability of artifacts, but also their semantic consistency and executable integrity across disciplines and lifecycle stages.

Under this envisioned framework, system models are progressively extended and enriched as the product transitions through lifecycle stages. Rather than treating models as static documentation, MBSE 2.0 encourages evolving them into dynamic digital assets that continuously guide design updates, testing validation, manufacturing decisions, and operational optimization. For instance, the construction of a digital twin is intentionally planned during early design phases, with models progressively updated through manufacturing and real-world operational feedback, ensuring that the virtual representation remains aligned with the evolving physical system for purposes such as predictive maintenance, performance optimization, and lifecycle cost management.

In summary, MBSE 2.0 promotes a lifecycle-wide model-driven paradigm, with the Model Bus proposed as a central infrastructure to facilitate consistent, dynamic model interaction across different phases. By advancing beyond traditional document-centric and fragmented toolchains, this approach aspires to create a broader, continuously evolving system model ecosystem that provides persistent decision support for system design, testing, production, operation, and evolution.

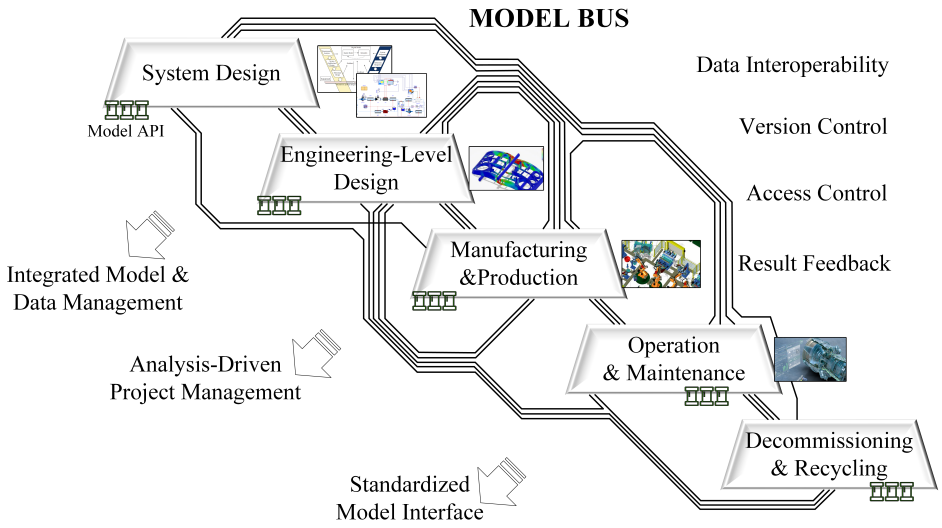

Figure 5. Illustration of the Model Bus concept within MBSE 2.0, highlighting its role as a central communication backbone that connects all stages of the system lifecycle (from initial system design to engineering, manufacturing, operation, and decommissioning). Much like a physical circuit bus, the Model Bus enables standardized model access, data interoperability, version control, and secure feedback mechanisms through a unified model API and standardized model interfaces. By supporting integrated model and data management, analysis-driven project control, and dynamic feedback loops, the Model Bus facilitates continuous model evolution and reuse across lifecycle phases, promoting cross-domain consistency and traceability.

4. Key Enabling Technologies of MBSE 2.0

Building on the conceptual framework outlined in Section 2, this section comprehensively summarizes the enabling technologies of MBSE 2.0 across ten main technical domains. An overview is provided in Figure 6.

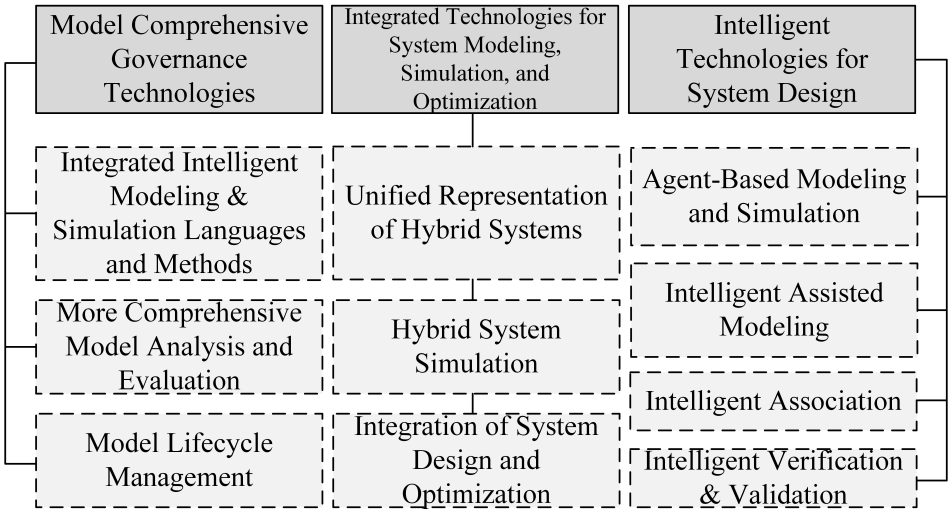

Figure 6. Key enabling technologies of MBSE 2.0 compared to MBSE.

4.1. Model Comprehensive Governance Technologies

The objects constructed in MBSE often possess characteristics such as requirement complexity, structural complexity, and relational complexity, which lead to a series of challenges in model construction, usage, and maintenance. Against this backdrop, model-related technologies play a pivotal role in model-based systems engineering.

However, research on fundamental model-related methods and theories remains insufficient. In response, this paper argues that the MBSE 2.0 framework necessitates deeper exploration of the theories, methods, and technologies governing the system model lifecycle. We collectively refer to these approaches as Model Comprehensive Governance Technologies. The following sections examine three crucial dimensions: model establishment, evaluation, and management, with particular emphasis on modeling and simulation languages and methods, model analysis and evaluation, and model lifecycle management. By strengthening these foundational pillars, Model Comprehensive Governance Technology not only underpins MBSE 2.0 but also enhances the reliability of system models throughout their lifecycle, guarantees high-quality modeling outcomes, and maximizes the overall value of system models.

4.1.1. Integrated Intelligent Modeling & Simulation Languages and Methods

Integrated intelligent modeling and simulation languages and methods form a critical foundation for achieving comprehensive system integration and enhanced intelligence. A central research objective is to develop unified representations of multi-stage, multi-type system models, thereby addressing the diverse and often complex needs of large-scale systems. This involves exploring robust metamodels tailored to various model types and establishing standardized modeling languages and methods that effectively support system design, simulation, verification, and optimization.

SysML v2 represents a significant step forward in standardizing system architecture modeling, offering enhanced semantic precision and extensibility. However, it remains primarily focused on structural and behavioral representations of system architectures. Modeling constructs for simulation semantics or design optimization remain loosely coupled and rely on external tools or integration frameworks. As such, further innovation is needed to develop unified modeling paradigms that can natively capture and interrelate system architecture, simulation, and optimization models within a coherent semantic layer.

With the growing prevalence of intelligent and adaptive systems, new modeling languages and methods must be devised to reflect the dynamic, learning-driven, and context-aware nature of such systems, thus supporting the next generation of AI-empowered system engineering.

4.1.2. More Comprehensive Model Analysis and Evaluation

The analysis and evaluation of models refer to methods and technologies for the verification, assessment, and analysis of indicators at various stages of the model's full lifecycle. This includes the validation and verification (VV&A) of model credibility, model maturity assessment, and other related activities. Model analysis and evaluation techniques ensure the reliability and validity of models, which is crucial for promoting continuous improvement in system design. Moreover, in MBSE 2.0, the rating and testing of model maturity should be developed into more standardized and executable methodologies, integrating AI and simulation technologies for comprehensive evaluation, thereby facilitating effective application in system design practices. Research on model quantitative analysis, including model complexity analysis, cost-benefit analysis, and risk analysis of model construction, use, reuse, and management, as well as model uncertainty assessment (aimed at supporting metric evaluation by quantifying the uncertainties of model-related variables), should also be further expanded to ultimately form a comprehensive model analysis and evaluation technology system.

4.1.3. Model Lifecycle Management in MBSE 2.0

Model Lifecycle Management (MLM) refers to the governance processes and management activities associated with models, aimed at creating, reading, updating, deleting, and

maintaining models from various sources over time. It also involves managing all data related to models, tools, and analytical outcomes to ensure consistent representation and updates to system models in development, thereby ensuring their effectiveness and maintainability throughout their lifecycle. This includes key capabilities such as configuration management, variant handling, branching and merging, model traceability, and access control, as emphasized in existing MBSE practices. A key research area in MLM in MBSE 2.0 is the Model Bus, which forms the foundation for horizontal integration and interaction across different stages of model development. By effectively connecting lifecycle data and models through the Model Bus, the full value of MBSE 2.0 can be realized. Another critical focus is model calibration and evolution, which ensures vertical adjustments to models over time. Digital thread technology also plays an essential role in MLM, enabling models to be tracked and managed across all stages and ensuring a single trusted data source to maintain consistency between system models and the actual state of the product or system. Furthermore, model modularization, servitization, and reuse are crucial research areas for the widespread adoption of model-based engineering principles.

4.2. Integrated Technologies for System Modeling, Simulation, and Optimization

In current MBSE practices, engineers are required to use multiple languages and tools, linking data layers to accomplish system modeling, simulation validation, and system optimization. However, heterogeneity and interoperability issues among different modeling tools pose cost and time risks for building an efficient toolchain in the R&D process. Additionally, limitations in models, tools, and methodologies often result in independently conducted modeling, simulation, and optimization processes. This separation prevents timely simulation validation of system designs and restricts optimization to local areas, thereby limiting design efficiency and quality. These challenges result in high barriers, lengthy cycles, and low efficiency in the current system design work.

To address these issues, the integrated technology for system modeling, simulation, and optimization aims to provide a unified model that supports comprehensive descriptions of system architecture and physical properties and can be directly used for simulation. Based on this unified model, global optimization of system design is achievable. Specifically, the integrated technology encompasses three aspects: unified representation of hybrid systems, hybrid system simulation, and integration of system design and optimization.

4.2.1. Unified Representation of Hybrid Systems

System architecture models, typically represented by logical and functional components, generally exhibit discrete behavioral characteristics. In contrast, a system's physical properties often exhibit continuous behaviors across multiple physical domains, sometimes incorporating dynamic behaviors of discrete events. The core challenge of unified representation between system architecture and simulation models lies in creating a cohesive description for hybrid systems with both continuous and discrete behaviors. Research in unified hybrid system representation mainly addresses coupling mechanisms for discrete events and continuous time, multi-physics hybrid modeling, and the integration of causal and acausal models. Achieving unified representation for hybrid systems requires substantial innovation in modeling languages and tools. The primary challenge lies in effectively integrating multi-physics domains and discrete logic systems into a unified meta-model framework. Additionally, it involves developing a universal model that can describe the behaviors of these heterogeneous systems while ensuring adaptability and flexibility across diverse application scenarios. This process is complex and demanding, involving deep cross-disciplinary exploration. Unified representation of hybrid systems

forms the foundation of integrated system modeling, simulation, and optimization, playing a crucial role in advancing the integration of system design and simulation.

4.2.2. Hybrid System Simulation

Hybrid system simulation aims to support the deep integration of system design and simulation. It refers to the comprehensive simulation process for systems that contain both discrete and continuous behaviors. This simulation must handle continuous dynamic processes within physical domains (such as mechanical, electrical, and hydraulic) along with the logical transitions of discrete events, while also accommodating interactions between causal and acausal systems. Hybrid system simulation requires research into multiple areas, including co-simulation algorithms for discrete events and continuous time, hybrid simulation frameworks for causal and acausal systems, and event handling and scheduling strategies. The focus is on overcoming challenges in modeling language compilation, integration of heterogeneous solvers, and balancing simulation accuracy with performance. As a key component for achieving integrated system modeling and simulation, hybrid system simulation is essential for enabling global system optimization. By employing hybrid system simulation techniques, model validation can be effectively supported throughout the design process, achieving timely design, simulation, and validation, thereby facilitating efficient design iteration.

4.2.3. Integration of System Design and Optimization

The integration of system design and optimization combines the design process with optimization techniques, aiming to dynamically adjust and improve system designs and key parameters through simulation and optimization algorithms to enhance performance and efficiency. Traditionally, system optimization has been a localized adjustment performed after the design is mostly finalized. The core of integrating system design and optimization lies in embedding the optimization model, including design variables, constraints, and objectives, into the design process, thereby ensuring that optimization spans the entire design cycle. This integration relies on automated methods to continuously feed optimization results back into the design model. Several challenges must be addressed to achieve effective integration. First, it is essential to coordinate global and local optimization across different design stages, ensuring that critical parameters can be dynamically adjusted to support early-stage architectural comparisons and later-stage global parameter refinement. Second, transforming complex design problems that span multiple physical domains and variables into solvable optimization models is crucial for effectively capturing key parameters and constraints. Additionally, establishing a feedback mechanism for optimization results is needed to automate the incorporation of optimization outcomes into the design model, ensuring real-time updates and improvements in the design.

4.3. Intelligent Technologies for System Design

As the scale and complexity of intelligent products grow, there is an increasing demand for intelligent technology in system design. On one hand, with advances in artificial intelligence, the need for modeling intelligent systems, subsystems, and intelligent behaviors has grown significantly. Traditional system architectures and model representations often lack adaptability, making it challenging to address dynamic environmental changes or evolving intelligence-related requirements, thereby impacting system stability and reliability. On the other hand, the inconsistency in models and tools across stages—such as requirements analysis, functional analysis, and design and verification—necessitates highly specialized engineers to bridge the gaps at each stage. The design process largely relies on manual inputs from engineers, with models constrained by individual expertise, making

innovative design difficult. This manual approach often results in inefficiencies, high labor costs, limited traceability, and a lack of intelligent automation in the design process.

Within MBSE 2.0, intelligent technologies for system design refer to the application of next-generation artificial intelligence within systems engineering. This includes agent-based modeling and simulation, intelligent assisted modeling, intelligent association, and intelligent validation. By leveraging intelligent technologies, the development of intelligent systems can accelerate design workflows, reduce labor costs, and enhance innovation capabilities, providing significant support for advanced system design.

4.3.1. Agent-Based Modeling and Simulation

Agent-based modeling and simulation (ABMS) technology is employed to build and simulate intelligent systems or components with autonomous behaviors and interactions within system design. This approach defines agents' interactive and learning behaviors, using event-driven transition mechanisms to represent their responses across varying scenarios. ABMS enables researchers and developers to create complex agent models and simulate their performance across diverse environments and contexts. However, the technique faces several challenges, including managing the complexity of agent interactions, handling diverse learning patterns, maintaining synchronization across parallel simulations, and ensuring model generalizability and interpretability. ABMS facilitates the design and testing of systems with advanced intelligent behaviors, such as autonomous decision-making, learning, optimization, and prediction. By simulating these agent-based models, it becomes possible to explore and validate intelligent behaviors under safe, controlled conditions, thereby ensuring their effectiveness and reliability upon deployment.

4.3.2. Intelligent Assisted Modeling

Intelligent Assisted Modeling (IAM) technology leverages new generation AI, particularly generative AI, to enable system generative design across various phases, including requirement analysis, functional and logical modeling, and system simulation model construction. In requirement analysis, IAM processes unstructured inputs to generate structured requirements and predicts potential gaps by referencing historical data. For functional and logical design, it autonomously constructs architecture and behavior models based on requirements and engineer guidance, compares alternative solutions, and ensures alignment with best practices. In system simulation, IAM generates unit designs, detailed specifications, and interface definitions based on architectural standards. Key challenges include the need for text-based model representations to exploit generative AI's capabilities, the integration of constraint verification tools, and the development of high-quality system model corpora. By shifting from manual to partially intelligent design, IAM empowers engineers to explore beyond conventional thinking, significantly enhancing design efficiency and innovation.

4.3.3. Intelligent Association

Intelligent association technology applies next-generation AI to build mapping relationships throughout the system design process. It captures and aligns relationships and key parameters across design stages, supporting tasks such as requirement–architecture alignment and architecture–simulation synchronization, especially during design changes. For example, by linking requirements to system architecture, it identifies how requirement changes impact design; by connecting architecture to simulation models, it ensures that simulations reflect the latest designs. The main challenge lies in accurately mapping parameters and logical dependencies, which are often implicit and hard to express. Overcoming this requires flexible mapping methods and the integration of technologies like knowledge graphs and large generative models to automatically discover and construct associations.

By interconnecting models, parameters, and domain knowledge, intelligent association enables global change management, efficient information retrieval, and supports iterative system model evaluation and design evolution.

Unlike traditional traceability, which typically depends on static, manually defined links between artifacts, intelligent association leverages AI technologies such as knowledge graphs and large language models to automatically infer, update, and reason over inter-model relationships. This enables context-aware, scalable, and dynamic mappings that support change impact analysis, iterative design refinement, and real-time decision support. By interconnecting models, parameters, and domain knowledge, intelligent association facilitates global change management, efficient information retrieval, and intelligent system model evolution.

4.3.4. Intelligent Verification & Validation

Intelligent verification and validation technologies leverage AI in combination with integrated modeling, simulation, and intelligent association techniques to enable automated, adaptive, and insightful system-level evaluation. These technologies are essential for ensuring both the correctness (verification) and fitness-for-purpose (validation) of complex, intelligent systems. A key capability is the use of AI-driven frameworks and automated scripts to support seamless integration, data exchange, and execution of simulation-based testing across multi-domain tools. This includes automating tasks such as parameter configuration, simulation orchestration, and result aggregation, which are traditionally time-consuming and prone to human error. In terms of verification, AI can assist in the automated generation of test cases and procedures, ensuring broad coverage of system behaviors and reducing the manual workload. For validation, simulation outputs are used to assess whether the system meets intended functional and performance objectives under realistic conditions. Importantly, intelligent validation incorporates causal analysis and issue diagnosis, allowing for a deeper understanding of model deviations and failure modes. Several challenges remain: achieving tool compatibility and semantic consistency is critical to forming a coherent modeling–simulation–testing pipeline. Moreover, reverse inference from simulation outcomes (identifying root causes of anomalies) requires advanced AI algorithms and explainable data analysis models. This highlights the importance of integrating intelligent association methods to enable effective issue localization and resolution.

5. Preliminary Efforts Toward MBSE 2.0 Implementation

By integrating the key enabling technologies of MBSE 2.0, its various features can reinforce and stimulate one another, fostering innovative applications across critical stages of the product lifecycle. Nevertheless, achieving the full vision of MBSE 2.0 still requires substantial effort and long-term commitment. This section presents several preliminary explorations conducted by the authors' team, aimed at demonstrating and validating some of the core ideas underlying the MBSE 2.0 framework.

5.1. An Integrated Modeling and Simulation Language: X Language

To explore the core ideas of MBSE 2.0, our team has developed X Language [74], a prototype integrated modeling and simulation language aimed at integrating system architecture modeling and executable simulation. The basic X language architecture is shown in Figure 7. Traditional MBSE often relies on fragmented toolchains, where architecture is specified in SysML-like languages and simulation is performed separately using tools such as Simulink or Modelica. This fragmentation leads to poor traceability and increased manual effort during model translation.

X Language addresses this gap by providing a unified syntax and execution framework for defining system structures, behaviors, and constraints. It supports both discrete-event and continuous-time modeling, enabling the representation of control logic and physical dynamics within a single language. Its modular, declarative structure allows for hierarchical system decomposition and supports early-stage simulation. Technically, X Language is implemented with a virtual simulation engine that enables hybrid model execution and supports introspection and traceability. It also allows integration with co-simulation tools and external solvers, making it extensible for multi-domain applications.

We have applied X Language in several test cases, such as aircraft electrical systems [75], civil aircraft take-off scenarios [76]. These trials show that such a unified language can streamline modeling workflows and support early validation—both key goals of MBSE 2.0. While still under development, X Language reflects an early step toward closing the gap between system specification and behavioral analysis, offering a technical path to realize integrated and intelligent modeling frameworks.

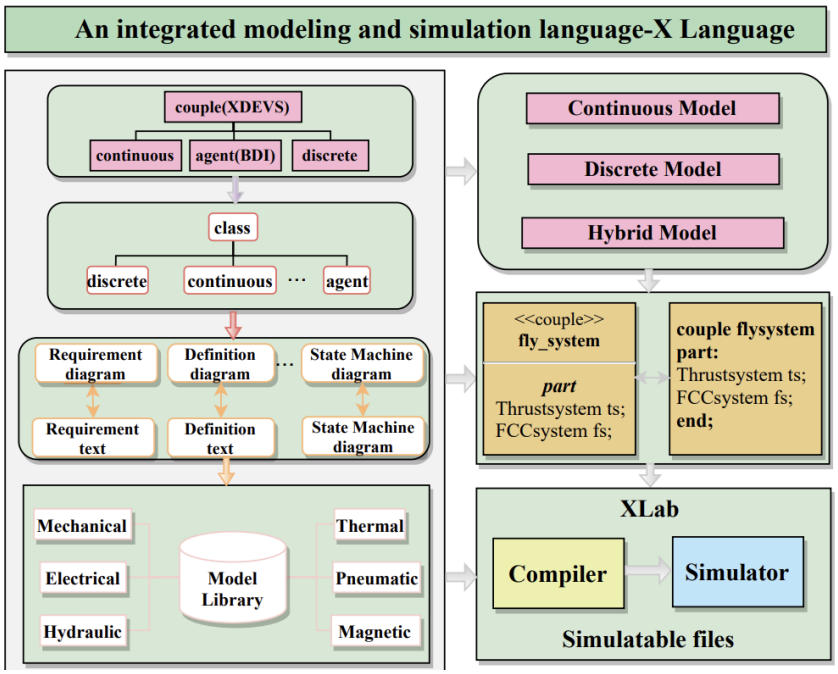

Figure 7. Schematic diagram of X language. X language facilitates the modeling of continuous, discrete, and hybrid systems, and supports model compilation and simulation through its development environment, XLab. This figure is adapted from [77].

5.2. Generative Design Based on X Language

Building upon the unified modeling foundation of X Language, our team further explored the use of large language models to support generative system design [77]. Specifically, we proposed a framework that leverages LLMs to automatically generate system modeling code in X Language from high-level natural language inputs. This approach aims to reduce the manual workload involved in system modeling and bridge the cognitive gap between domain experts and formal modeling tools.

The proposed framework integrates an LLM with a structured prompt framework that includes system design patterns, functional requirements, and modeling templates. Given a textual description of a system—such as objectives, components, and constraints—the model generates executable X Language code that captures both the structural and behavioral aspects of the system. This process supports rapid prototyping and design-space exploration at the early stages of system development.

This generative approach represents a promising direction for AI-assisted system modeling and is aligned with one of the core pillars of MBSE 2.0. By leveraging large language models in conjunction with domain-specific languages like X Language, it enables intelligent co-design workflows that improve accessibility, accelerate model creation, and reduce entry barriers for non-expert users. More importantly, under a unified modeling paradigm that integrates system architecture and physical simulation semantics, such generative mechanisms facilitate a seamless transition from high-level requirements to executable and verifiable simulation models. This ensures better model consistency across design layers and significantly shortens the iteration cycle between conceptual design and behavioral validation.

6. Conclusions

This paper analyzes the current limitations of MBSE practices and identifies several emerging trends toward more integrated, intelligent, and lifecycle-oriented development. Based on these observations, the concept of MBSE 2.0 is proposed to capture key directions in future systems engineering methodology. The framework outlines potential paths for enhancing modeling capabilities, improving cross-domain integration, and supporting intelligent analysis and evaluation.

While this paper outlines the conceptual potential of MBSE 2.0, it is important to acknowledge several practical challenges that may impede its widespread adoption. First, the implementation of AI-integrated MBSE approaches may involve significant upfront costs, including investment in new tools, model repositories, and training infrastructure. Second, the absence of mature standards for intelligent model representation, reasoning, and governance can hinder interoperability and long-term maintainability. These limitations suggest that the evolution toward MBSE 2.0 must be pursued progressively, with pilot applications, toolchain co-development, and community-driven standardization efforts serving as essential steps forward.

Implementing MBSE 2.0 as a systems engineering framework involves not only overcoming technical challenges but also embracing a new philosophy that spans multiple organizational dimensions. A shift towards fostering an innovative and efficient engineering culture is vital for achieving effective system design. Leading-edge companies in the technology and manufacturing sectors often embrace flat management structures and cross-departmental collaboration, which enhances innovation and efficiency. In contrast, rigid organizational hierarchies, excessive rules, and complex processes can hinder progress in systems engineering. An effective systems engineering culture balances creativity with disciplined engineering methods, ensuring teams maintain a holistic approach to problem-solving and deliver complete, integrated solutions.

Looking to the future, significant breakthroughs are needed in talent development and organizational structuring, especially in finding a sustainable balance between cost and return. These gradual advancements will pave the way for the full realization of MBSE 2.0 and its transformative potential in systems engineering.

At the same time, we acknowledge that the continued advancement of MBSE requires the integration of more rigorous and systematic methodologies. In particular, there remains a need to extract mathematically grounded structures (such as unified decision models, optimization principles, and preference representations) from the inherent complexity of real-world systems. While this paper emphasizes engineering practices and the orchestration of enabling technologies, we believe that future developments will benefit from a tighter coupling between practical modeling frameworks and formal theoretical foundations. We hope our work can inspire not only methodological innovations but also deeper theoretical inquiry into the mathematical essence of systems engineering.

Author Contributions: All authors contributed to the study conception and design. L.Z.: Methodology, Writing—Review and editing, Funding acquisition, Supervision. Z.C.: Methodology, Writing—Original draft, Writing—Review and editing, Conceptualization, Supervision. L.R., Y.L. and M.J.D.: Supervision, Writing—Review and editing. W.C., Y.Z. (Yuteng Zhang), Y.Z. (Yuqing Zeng) and P.G.: Visualization, Investigation, Writing—Editing. All authors have read and agreed to the published version of the manuscript.

Funding: This work is supported by the National Key R&D Program of China (2023YFB3308200), National Natural Science Foundation of China (Grant Nos. 62373026), Natural Science Foundation of Beijing (Grant Nos. L241018), Beihang World TOP University Cooperation Program, and the Academic Excellence Foundation of BUAA for Ph.D. Students.

Data Availability Statement: The data that support the findings of this study are available upon request from the corresponding author.

Conflicts of Interest: Author M. Jamal Deen was employed by the company AI Atlas Inc. The remaining authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

References

1. Henderson, K.; Salado, A. Value and benefits of model-based systems engineering (MBSE): Evidence from the literature. *Syst. Eng.* **2021**, *24*, 51–66. [CrossRef]
2. Montgomery, P.R. Model-Based System Integration (MBSI)–Key Attributes of MBSE from the System Integrator's Perspective. *Procedia Comput. Sci.* **2013**, *16*, 313–322. [CrossRef]
3. Manjón, J.V.G. Understanding Innovation And Technology: Key Definitions and Concepts. In *Technological Innovation Strategy and Management*; World Scientific Book Chapters; World Scientific: Singapore, 2020; pp. 1–31.
4. Tischner, U.; Charter, M. Sustainable product design. In *Sustainable Solutions*; Routledge: Abingdon, UK, 2017; pp. 118–138.
5. Lu, J.; Wen, Y.; Liu, Q.; Gürdür, D.; Törngren, M. MBSE applicability analysis in Chinese industry. In Proceedings of the INCOSE International Symposium, Washington, DC, USA, 7–11 July 2018; Wiley Online Library: Hoboken, NJ, USA, 2018; Volume 28, pp. 1037–1051.
6. D'Souza, A.; Thota, P. Configuration Management for Model Based Systems Engineering—An Example from the Aerospace Industry. *INSIGHT* **2023**, *26*, 60–66. [CrossRef]
7. Gough, K.M.; Phojanamongkolkij, N. Employing model-based systems engineering (MBSE) on a NASA aeronautic research project: A case study. In Proceedings of the 2018 Aviation Technology, Integration, and Operations Conference, Atlanta, GA, USA, 25–29 June 2018; p. 3361.
8. Tang, X.; Yan, W.; Song, G. Model-Based Design Method and Practice of Avionics System Architecture in Civil Aircraft. In *Complex Systems Design & Management, Proceedings of the 14th International Conference on Complex Systems Design & Management, Beijing, China, 30–31 October 2023*; Springer: Berlin/Heidelberg, Germany, 2023; pp. 189–199.
9. Lu, J. *Research Survey on Model-Based Systems Engineering Tool-Chain*; Technical Report; School of Industrial Engineering and Management: Stockholm, Sweden, 2019. Available online: https://www.diva-portal.org/smash/get/diva2:1313985/FULLTEXT01.pdf (accessed on 8 July 2025).
10. De Saqui-Sannes, P.; Vingerhoeds, R.A.; Garion, C.; Thirioux, X. A taxonomy of MBSE approaches by languages, tools and methods. *IEEE Access* **2022**, *10*, 120936–120950. [CrossRef]
11. Ma, J.; Wang, G.; Lu, J.; Vangheluwe, H.; Kiritsis, D.; Yan, Y. Systematic literature review of MBSE tool-chains. *Appl. Sci.* **2022**, *12*, 3431. [CrossRef]
12. Zeigler, B.P.; Mittal, S.; Traore, M.K. MBSE with/out Simulation: State of the Art and Way Forward. *Systems* **2018**, *6*, 40. [CrossRef]
13. Pandolf, J. Investigation of Model-Based Systems Engineering Integration Challenges and Improvements. Ph.D. Thesis, Massachusetts Institute of Technology, Cambridge, MA, USA, 2023.
14. Estefan, J.A.; Weilkiens, T. MBSE methodologies. In *Handbook of Model-Based Systems Engineering*; Springer: Berlin/Heidelberg, Germany, 2023; pp. 47–85.
15. Morkevicius, A.; Aleksandraviciene, A.; Mazeika, D.; Bisikirskiene, L.; Strolia, Z. MBSE grid: A simplified SysML-based approach for modeling complex systems. In Proceedings of the INCOSE International Symposium, Adelaide, Australia, 15 July 2017; Wiley Online Library: Hoboken, NJ, USA, 2017; Volume 27, pp. 136–150.
16. Kianzad, S.; Seifi, H. MagicDraw: Haptic-Assisted One-Line Drawing with Shared Control. In Proceedings of the 37th Annual ACM Symposium on User Interface Software and Technology, Pittsburgh, PA, USA, 13–16 October 2024; pp. 1–3.

17. Guo, J.; Wang, G.; Lu, J.; Ma, J.; Törngren, M. General modeling language supporting model transformations of mbse (part 2). In Proceedings of the INCOSE International Symposium, Cape Town, South Africa, 18–23 July 2020; Wiley Online Library: Hoboken, NJ, USA, 2020; Volume 30, pp. 1460–1473.
18. Zhang, L.; Ye, F.; Laili, Y.; Xie, K.; Gu, P.; Wang, X.; Zhao, C.; Zhang, X.; Chen, M. X language: An integrated intelligent modeling and simulation language for complex products. In Proceedings of the 2021 Annual Modeling and Simulation Conference (ANNSIM), San Diego, CA, USA, 19–22 July 2021; pp. 1–11.
19. *ISO/IEC/IEEE 15288:2023*; Systems and Software Engineering—System Life Cycle Processes. ISO: Geneva, Switzerland 2023.
20. *ISO/IEC/IEEE 24641:2023*; Systems and Software Engineering—Methods and Tools for Model-Based Systems and Software Engineering. ISO: Geneva, Switzerland 2023.
21. Vaicenavičius, J.; Wiklund, T.; Kavolis, D.; Draukšas, S.; Kalkauskas, A.; Vaicenavičius, R. SysIDE: SysML v2 textual editing and analysis system: Overview and applications. *CEAS Space J.* **2025**, 1–7. [CrossRef]
22. Lameh, J.; Dubray, A.; Jankovic, M. Variability in complex product/system design: Case study in automotive industry. Proceedings of the International Design Conference, Dubrovnik, Croatia, 20–23 May 2024; Design Society: Glasgow, UK, 2024; Volume 4, pp. 2635–2644.
23. Berschik, M.C.; Schumacher, T.; Laukotka, F.N.; Krause, D.; Inkermann, D. Mbse Within the Engineering Design Community—An Exploratory Study. In Proceedings of the Design Society, Cavtat, Dubrovnik, Croatia, 20–23 May 2024; Volume 3, pp. 2595–2604.
24. Blott, J.; Buchholz, C. *Challenges of Implementing MBSE in Industry: A Tool Vendor Experience*; Institute of Technology: Ilmenau, Germany, 2023.
25. Bachelor, G.; Brusa, E.; Ferretto, D.; Mitschke, A. Model-based design of complex aeronautical systems through digital twin and thread concepts. *IEEE Syst. J.* **2019**, *14*, 1568–1579. [CrossRef]
26. Pavalkis, S. Towards Industrial Integration of MBSE into PLM for Mission-Critical Systems. In Proceedings of the INCOSE International Symposium, Bangalore, India, 9–11 November 2016; Wiley Online Library: Hoboken, NJ, USA, 2016; Volume 26, pp. 2462–2477.
27. Gu, P.; Chen, Z.; Zhang, L.; Zhang, Y.; Xie, K.; Zhao, C.; Ye, F.; Tao, Y. X-SEM: A modeling and simulation-based system engineering methodology. *J. Manuf. Syst.* **2024**, *74*, 198–221. [CrossRef]
28. Wise, R.; Zimmer, R. Empowering Model-Based Systems Engineering Through Metamodeling. In Proceedings of the INCOSE International Symposium, Dublin, Ireland, 2–6 July 2024; Wiley Online Library: Hoboken, NJ, USA, 2024; Volume 34, pp. 1616–1630.
29. Ding, J.; Reniers, M.; Lu, J.; Wang, G.; Feng, L.; Kiritsis, D. Integration of modeling and verification for system model based on KARMA language. In Proceedings of the 18th ACM SIGPLAN International Workshop on Domain-Specific Modeling, Chicago, IL, USA, 18 October 2021; pp. 41–50.
30. Friedenthal, S. Future Directions for MBSE with SysML v2. In Proceedings of the MODELSWARD, Lisbon, Portugal, 19–21 February 2023; pp. 5–9.
31. Cederbladh, J.; Berardinelli, L.; Bruneliere, H.; Cicchetti, A.; Dehghani, M.; Di Sipio, C.; Miranda, J.; Rahimi, A.; Rubei, R.; Suryadevara, J. Towards Automating Model-Based Systems Engineering in Industry—An Experience Report. In Proceedings of the 2024 IEEE International Systems Conference (SysCon), Montreal, QC, Canada, 15–18 April 2024; pp. 1–8.
32. DeHart, J.K. Leveraging Large Language Models for Direct Interaction with SysML v2. In Proceedings of the INCOSE International Symposium, Dublin, Ireland, 2–6 July 2024; Wiley Online Library: Hoboken, NJ, USA, 2024; Volume 34, pp. 2168–2185.
33. Zhang, D.; Li, W.; Wang, C.; Li, J. Simulation research on breakdown diagnosis based on least squares support vector machine optimized by simulated annealing genetic algorithm. *Int. J. Model. Simul. Sci. Comput.* **2024**, *15*, 2350046. [CrossRef]
34. INCOSE. *INCOSE Systems Engineering Handbook*; John Wiley & Sons: Hoboken, NJ, USA, 2023.
35. Motamedian, B. MBSE applicability analysis. *Int. J. Sci. Eng. Res.* **2013**, *4*, 7.
36. Madni, A.M.; Sievers, M. Model-based systems engineering: Motivation, current status, and research opportunities. *Syst. Eng.* **2018**, *21*, 172–190. [CrossRef]
37. Hoffmann, H.P. SysML-based systems engineering using a model-driven development approach. In Proceedings of the INCOSE International Symposium, Orlando, FL, USA, 9–13 July 2006.
38. Estefan, J.A. Survey of model-based systems engineering (MBSE) methodologies. *Incose Mbse Focus Group* **2007**, *25*, 1–12.
39. Douglass, B. *The Harmony Process [I-Logix White Paper]*; I-Logix: Burlington, MA, USA, 2005.
40. Yang, Z.; Du, H.; Liu, Y.; Liu, R.; Liu, Y. Use the harmony-SE approach to extend the advantages of MBSE. In Proceedings of the 2021 IEEE 16th Conference on Industrial Electronics and Applications (ICIEA), Chengdu, China, 1–4 August 2021; pp. 223–227.
41. Lykins, H.; Friedenthal, S.; Meilich, A. Adapting UML for an Object Oriented Systems Engineering. In Proceedings of the INCOSE Cheasepeake Chapter 2001 Meeting, Minneapolis, MN, USA, 16–20 July 2001.
42. Wolfrom, J. *Model-Based System Engineering (MBSE) Using the Object-Oriented Systems Engineering Method (OOSEM)*; The Johns Hopkins University Applied Physics Laboratory: Laurel, MD, USA, 2011.

43. Kruchten, P. Tutorial: Introduction to the rational unified process®. In Proceedings of the 24th International Conference on Software Engineering (ICSE), Orlando, FL, USA, 19–25 May 2002; p. 703.
44. Kruchten, P. *The Rational Unified Process: An Introduction*; Addison-Wesley Professional: Boston, MA, USA, 2004.
45. Hanssen, G. K.; Bjørnson, F. O.; Westerheim, H. Tailoring and introduction of the rational unified process. In Proceedings of the 14th European Conference on Software Process Improvement (EuroSPI), Potsdam, Germany, 26–28 September 2007; pp. 7–18.
46. Long, J.E. Systems Engineering (SE) 101. In *CORE®: Product & Process Engineering Solutions, Vitech Training Materials*; Vitech Corporation: Vienna, VA, USA, 2000.
47. Long, J.E. A concurrent methodology for the system engineering design process. *System* **1994**, *3*, 3.
48. Ingham, M. State Analysis Overview: What PSEs ought to know. In *Briefing Slides (Internal Document)*; Jet Propulsion Laboratory, California Institute of Technology: Pasadena, CA, USA, 2006.
49. Kordon, M.; Wall, S.; Stone, H.; Blume, W.; Skipper, J.; Ingham, M.; Neelon, J.; Chase, J.; Baalke, R.; Hanks, D.; et al. Model-based engineering design pilots at jpl. In Proceedings of the 2007 IEEE Aerospace Conference, Big Sky, MT, USA, 3–10 March 2007; pp. 1–20.
50. Dori, D. *Object-Process Methodology: A Holistic Systems Paradigm, with CD-ROM*; Springer Science & Business Media: Berlin/Heidelberg, Germany, 2002.
51. Dori, D.; Reinhartz-Berger, I. An OPM-based metamodel of system development process. In Proceedings of the International Conference on Conceptual Modeling, Tampere, Finland, 7–11 October 2002; pp. 105–117.
52. Dori, D. Object-process methodology. In *Knowledge Management: Concepts, Methodologies, Tools, and Applications*; IGI Global Scientific Publishing: Hershey, PA, USA, 2008; pp. 421–434.
53. Xia, Z.; Fu, R. Study on Space-Borne Remote Sensing Imaging System by MagicGrid Methodology. In Proceedings of the 7th International Conference on Signal and Information Processing, Networking and Computers (ICSINC), Rizhao, China, 21–23 September 2020; pp. 280–287.
54. Aleksandraviciene, A.; Morkevicius, A.M. *MagicGrid® Book of Knowledge: A Practical Guide to Systems Modeling Using MagicGrid from Dassault Systèmes*; Dassault Systèmes: Vélizy-Villacoublay, France, 2021.
55. Schumacher, T.; Inkermann, D. Heterogeneous models to support interdisciplinary engineering-mapping model elements of SysML and CAD. *Procedia CIRP* **2022**, **109**, 653–658. [CrossRef]
56. Pohl, K.; Hönninger, H.; Achatz, R.; Broy, M. *Model-Based Engineering of Embedded Systems: The SPES 2020 Methodology*; Springer: Berlin/Heidelberg, Germany, 2012.
57. Spütz, K.; Jacobs, G.; Zerwas, T.; Konrad, C. Modeling language for the function-oriented development of mechatronic systems with motego. *Forsch. Ingenieurwesen* **2023**, *87*, 387–398. [CrossRef]
58. Moeser, G.; Grundel, M.; Weilkiens, T.; Kümpel, S.; Kramer, C.; Albers, A. Modellbasierter mechanischer Konzeptentwurf: Ergebnisse des FAS4M-Projektes. *Tag Syst. Eng.* **2016**, 419–428.
59. Albers, A.; Moeser, G.; Revfi, S. Synergy effects by using sysml models for the lightweight design method "extended target weighing approach". *Procedia CIRP* **2018**, *70*, 434–439. [CrossRef]
60. Drave, I.; Rumpe, B.; Wortmann, A.; Berroth, J.; Hoepfner, G.; Jacobs, G.; Spuetz, K.; Zerwas, T.; Guist, C.; Kohl, J. Modeling mechanical functional architectures in SysML. In Proceedings of the 23rd ACM/IEEE International Conference on Model Driven Engineering Languages and Systems, Montreal, QC, Canada, 18–23 October 2020; pp. 79–89.
61. Hoepfner, G.; Nachmann, I.; Zerwas, T.; Berroth, J.K.; Kohl, J.; Guist, C.; Rumpe, B.; Jacobs, G. Towards a holistic and functional model-based design method for mechatronic cyber-physical systems. *J. Comput. Inf. Sci. Eng.* **2023**, *23*, 051001. [CrossRef]
62. Jacobs, G.; Konrad, C.; Berroth, J.; Zerwas, T.; Höpfner, G.; Spütz, K. Function-oriented model-based product development. In *Design Methodology for Future Products: Data Driven, Agile and Flexible*; Springer: Cham, Switzerland, 2022; pp. 243–263.
63. Rangel, D.; Pavalkis, S.; Yakimenko, O. Integration of Cameo Systems Modeler with Simulink for Co-Orbital Engagement Mission Engineering. In Proceedings of the INCOSE International Symposium, Honolulu, HI, USA, 15–20 July 2023; Wiley Online Library: Hoboken, NJ, USA, 2023; Volume 33, pp. 704–723.
64. Hoffmann, H.P. Deploying model-based systems engineering with IBM® rational® solutions for systems and software engineering. In Proceedings of the 2012 IEEE/AIAA 31st Digital Avionics Systems Conference (DASC), Williamsburg, VA, USA, 14–18 October 2012; pp. 1–8.
65. Roques, P. MBSE with the ARCADIA Method and the Capella Tool. In Proceedings of the 8th European Congress on Embedded Real Time Software and Systems (ERTS 2016), Toulouse, France, 27–29 January 2016.
66. Xie, K.; Zhang, L.; Li, X.; Gu, P.; Chen, Z. SES-X: A MBSE methodology based on SES/MB and X language. *Information* **2022**, *14*, 23. [CrossRef]
67. Longshore, R.; Bell, R.; Madachy, R. *Leveraging Generative AI to Modify and Query MBSE Models*; Technical Report, Acquisition Research Program; 2024. Available online: https://dair.nps.edu/handle/123456789/5113 (accessed on 8 July 2025)
68. Patel, A.; Maheshwaran, Y.; Santhya, P. Easing adoption of model based system engineering with application of generative ai. In Proceedings of the 2024 IEEE Space, Aerospace and Defence Conference (SPACE), Karnataka, India, 22–23 July 2024; pp. 871–874.

69. Xu, T.; Moalla, N.; Bentaha, M.L.; Aktekin, H.; Agostinelli, C. AI-Integrated Framework for Enhancing High Level Architecture Design Across System Lifecycle Stages. In Proceedings of the 13th International Conference on Model-Based Software and Systems Engineering, Porto, Portugal, 26–28 February 2025.
70. Johns, B.; Carroll, K.; Medina, C.; Lewark, R.; Walliser, J. AI Systems Modeling Enhancer (AI-SME): Initial Investigations into a ChatGPT-enabled MBSE Modeling Assistant. In Proceedings of the INCOSE International Symposium, Dublin, Ireland, 2–6 July 2024; Wiley Online Library: Hoboken, NJ, USA, 2024; Volume 34, pp. 1149–1168.
71. Fuchs, J.; Helmerich, C.; Holland, S. Transforming System Modeling with Declarative Methods and Generative AI. In Proceedings of the AIAA SCITECH 2024 Forum, Orlando, FL, USA, 8–12 January 2024; p. 1054.
72. Göksidan, H.T.; Solakoğlu, E. How Product Lifecycle Management (PLM) Creates Value: An Evidence-Based Look at the. *Int. J. Appl.* **2025**, *10*, 1–8.
73. Gautier, R.H.; Lepez Da Silva Duarte, N.; Baker, A.; Mavris, D. Formulation of an Ontology-Based Digital Thread for Analysis and Decision Support Activities. In Proceedings of the AIAA SCITECH 2025 Forum, Orlando, FL, USA, 6–10 January 2025; p. 1092.
74. Zhang, L.; Ye, F.; Xie, K.; Gu, P.; Wang, X.; Laili, Y.; Zhao, C.; Zhang, X.; Chen, M.; Lin, T.; et al. An integrated intelligent modeling and simulation language for model-based systems engineering. *J. Ind. Inf. Integr.* **2022**, *28*, 100347. [CrossRef]
75. Gu, P.; Li, Y.; Wu, Z.; Chen, Z.; Xie, K.; Zhang, L. Research on Integrated Modeling and Simulation Method of Missile Electrical System Based on X Language. In Proceedings of the China Intelligent Networked Things Conference, Xi'an, China, 17–18 May 2024; pp. 219–230.
76. Gu, P.; Zhang, L.; Chen, Z.; Ye, J. Collaborative design and simulation integrated method of civil aircraft take-off scenarios based on X language. *J. Syst. Simul.* **2022**, *34*, 929–943.
77. Zhang, L.; Zhang, Y.; Niyato, D.; Ren, L.; Gu, P.; Chen, Z.; Laili, Y.; Cai, W.; Bruzzone, A. GenAI for Simulation Model in Model-Based Systems Engineering. *arXiv* **2025**, arXiv:2503.06422.

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.