International Journal of Science and Research (IJSR)
ISSN: 2319-7064
SJIF (2022): 7.942

Enhancing Systems Engineering and Control Development: The Power of Model-Based Approaches in Early State of Product Development

Roopak Ingole

Manager – Advanced Embedded Software*Corporate Research & Technology Cummins Inc.* Columbus IN, USA
Email: *roopak.ingole[at]cummins.com*

Abstract: ***Model-Based Systems Engineering (MBSE) and Model-Based Rapid Control Prototyping are methodologies that significantly enhance the design, development, simulation, and testing of complex systems and control strategies. This paper explores the integration of MBSE with Simulink, a platform widely recognized for its robust simulation and model-based design capabilities. We discuss the principles of MBSE, the methodology of rapid control prototyping, and the application of Simulink in these contexts. Additionally, we analyze the benefits, challenges, and potential future developments in the field. The adoption of Simulink in MBSE and control development accelerates the engineering process, reduces costs, and improves product quality through early validation of system designs and behavior during the early phases of productdevelopment.***

Keywords: Model-Based Systems Engineering, Rapid Control Prototyping, Simulink, complex systems, product development

1. Introduction

Model-Based Systems Engineering (MBSE) is an approach to systems engineering that focuses on creating and exploiting domain models as the primary means of information exchange between engineers, rather than on traditional document-based information exchange [1]. It involves the use of visual modeling techniques to articulate system requirements, design, analysis, verification, and validation activities beginning in the conceptual design phase and continuing throughout development and later life cycle phases [1].

Model-Based Rapid Control Prototyping refers to the use of models to speed up the process of developing control systems. This approach uses simulations to test and refine controls beforethey are implemented in prototype hardware. It reduces development time and increases the ability to diagnose system behaviors in response to changes in control strategies [2].

Simulink, The MathWorks tool, is extensively used for dynamic system modeling and simulation. It supports system- level design, simulation, automatic code generation, and continuous test and verification of embedded systems. The MathWorks recently integrated System Composer toolbox in their Simulink product make it an ideal tool for implementing MBSE and rapid control prototyping.

MBSE with SIMULINK

Model-Based Systems Engineering (MBSE) promotes the use of models rather than traditional document-based approaches for information exchange among engineers throughout the system lifecycle [1]. Simulink, with its powerfulsimulation and modeling capabilities, has launched System Composer [3] toolbox supporting MBSE. System Composer enables the specification and analysis of architectures for model-based systems engineering and software architecture modeling. With System Composer, we can allocate the requirements while refining an architecture model that can then be designed and simulated in Simulink. This section elaborates on how Simulink's System Composer enhances MBSE by facilitating system design, requirements engineering, and validation processes.

a) System Design and Architecture

The foundational stage of MBSE involves creating a robust system design and architecture that aligns with the project requirements. System Composer supports this stage by providing an intuitive graphical environment where systems engineers can construct detailed, functional models of the system's architecture. These models represent various system components, their interactions and stereotypes and can optimizethem in real-time [4].

b) Visual Modeling

System Composer is developed on Simulink engine that supports graphical design for building system models, which can include continuous-time, discrete-time, and hybrid systems.This capability allows engineers to visualize complex interactions and data flows within the system, enhancing the clarity and accuracy of the design. Engineers can simulate different configurations of system architecture to identify the most efficient design before any physical prototypes are developed.

c) Requirements Engineering

An essential aspect of MBSE is the accurate capture and management of system requirements. Both Simulink and System Composer integrates seamlessly with requirements management tools, like PTC RV&S, utilizing standardized ReqIF™ [5] specification, enabling a bidirectional traceability that ensures every design step is aligned with the specified requirements [4].

d) Requirements Traceability

Like Simulink, System Composer allows for linking system models directly to requirements documents. This linkage is

crucial for verifying that all system requirements are met and provides a clear trace from the model back to the individual requirements. This traceability helps in maintaining the consistency of the system design with the stakeholders' expectations and regulatory standards [4].

e) Verification and Validation

Each component and subsystem modeled in System Composer can be subjected to a series of tests to verify that they meet the required specifications at the system level. System Composer facilitates the simulation of models under various operational conditions to validate the system against its requirements. This process helps in identifying and rectifying potential issues early in the development phase, reducing the risk of costly changes during later stages [6].

f) System-Level Analysis

With System Composer, engineers can perform system-level analysis to assess the overall performance of the system. This includes analyzing the interaction between subsystems and their impact on the system's efficiency and effectiveness. System- level simulations help in optimizing the system architecture by identifying bottlenecks and redundancies [6].

Incorporating System Composer into Model-Based Systems Engineering transforms the approach to system design and development. It allows for a more structured, efficient, and accurate design process, from initial concept through to detailed design and verification. The visual and modular nature of System Composer models enhances both understanding and communication among project stakeholders, facilitating a collaborative and iterative design environment. By leveraging System Composer within MBSE, organizations can achieve faster development times, reduce costs, and improve overall system quality and reliability.

2. Model-Based Rapid Control Prototyping using SIMULINK

As authors described in their paper "Hardware Prototyping using FreeRTOS while developing AutoSAR compliant application software using Simulink" [7], Model-Based Rapid Control Prototyping (MB-RCP) utilizes models to accelerate the design and deployment of control systems, enhancing the ability to test and refine these systems before their physical implementation. Simulink, a pivotal tool in this domain, offers a comprehensive environment for designing, testing, and deploying control algorithms efficiently. This section delves into how Simulink facilitates rapid control prototyping through its simulation capabilities, automatic code generation, and support for Hardware-in-the-Loop (HIL) testing.

1) Development of Control Algorithms

Control systems are crucial for managing the behavior of dynamic systems in engineering applications like Engines & Powertrains. Developing robust control algorithms is essential for ensuring system stability, performance, and efficiency. Simulink provides a robust platform for developing these control algorithms through a visual programming environment.

2) Visual Environment and Block Libraries

Simulink's graphical interface allows engineers to drag and drop blocks to build control systems, making it accessible even for those with limited programming expertise. It offers an extensive library of predefined blocks that represent various mathematical and engineering functions, enabling the rapid construction and modification of complex control algorithms.

3) Simulation and Model Testing

One of the core advantages of using Simulink for control development is its simulation capability. Engineers can simulate the control model under various scenarios and input conditions to observe how it responds. This immediate feedback is vital for refining control strategies and ensuring that the controller performs as expected under different conditions.

4) Automatic Code Generation

A significant feature of Simulink that enhances the speed of control development is its ability to automatically generate code from the models. This feature bridges the gap between the design and implementation stages of control system development. Simulink's support for various languages and compliance with different standards like, A-SPICE, AutoSAR, MISRA, etc. makes it go-to tool for controls engineering community. Simulink can generate C, C++, and HDL code from the control models that can run on real hardware. This capability allows for rapid prototyping and testing, as the code generated is optimized for performance and is directly deployable to microcontrollers, FPGA and DSPs.

5) Hardware-in-the-Loop (HIL) Testing

HIL testing is crucial for validating control algorithms in a real-time scenario without the risk and cost associated with deploying them directly into live operational systems. In HIL setups, the control algorithm (running on a processor or an FPGA) interacts with a simulated model of the physical system implemented in Simulink. This setup tests the control algorithm's real-time performance and robustness against hardware failures or unexpected physical interactions. HIL testing provides critical feedback that can be used to iteratively refine the control model. It allows engineers to adjust and optimize control parameters and strategies based on real data, ensuring the control system is both effective and efficient before final deployment.

Model-Based Rapid Control Prototyping using Simulink streamlines the process of designing and implementing control systems during early phases of product design and development. By leveraging Simulink's capabilities in visual modeling, simulation, automatic code generation, and HIL testing, engineers can develop sophisticated control strategies that are both robust and adaptable to changing system requirements. This integrated approach not only accelerates the development cycle but also enhances the reliability and performance of control systems, making it a preferred choice in industries where control systems play a critical role.

3. Evaluation of MBSE along with RCP with SIMULINK

Building upon the foundational use of Mathworks' webinar, "Bridging Model-Based Systems Engineering and Model-Based Design" [8] our team embarked on a venture to explore and harness the potential of System Composer for enhancing our Rapid Control Prototyping (RCP) processes. This endeavor was aimed at refining our development strategy during the crucial research and early product development phases, where preliminary design decisions greatly influence the final productquality and efficacy.

A. Integration of System Composer into the Development Workflow

Our existing workflow heavily relied on Simulink for developing control systems, particularly with our recent move towards adopting AutoSAR for engine control development. Simulink was pivotal in crafting a robust software development workflow that aligned with research objectives while ensuring that our production tools and processes remained undisturbed. This dual focus was critical in maintaining continuity and efficiency across project phases.

1) Enhancing Rapid Control Prototyping

Previously, as outlined in specific research papers, we utilized AutoSAR Composer for the architectural design and implementation of control systems (Figure 1). This tool was instrumental in allowing us to quickly devise new control strategies and validate them on actual hardware with minimal delays. This phase of rapid prototyping is vital for iterating design enhancements and addressing potential functional issuesearly in the development cycle.

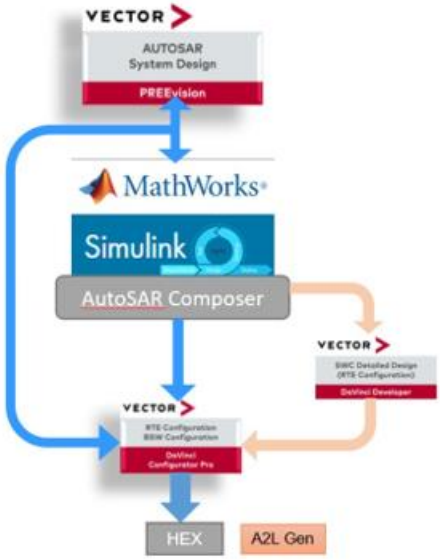

Figure 1: Rapid Control Prototyping workflow with AutoSAR

In our model-based rapid control prototyping workflow, system requirements were meticulously captured using PTC Integrity (RV&S), then exported in ReqIF [5] format and imported into Simulink Requirements. This process enabled precise requirement tracing at the algorithmic level. However, we recognized the need for a more integrated approach that could encompass comprehensive system analysis and multi- domain analysis, capabilities that were initially lacking.

2) System Composer's Role in Advancing MBSE and RCP

The introduction of System Composer sparked a shift in our approach. This tool promised to bridge the gaps we experienced, particularly in system analysis and integration. Our evaluation project centered on developing a Battery Management System (BMS), which served as an ideal platform to test the effectiveness of System Composer in a complex, multi-dimensional engineering environment.

3) Architectural and Component-Level Design

Using System Composer, we began by constructing a high-level model of the BMS architecture. We applied various parameterized stereotypes to components, such as CAN network latency and baud rate, and the number of cells per pack. These parameters were crucial for running detailed system analyses, which helped us understand the system's robustness against parameter variations and assisted in refining systemrequirements (Figure 2).

At a more granular level, we further designed the BMS's component architecture within System Composer. This involved utilizing Simulink and AutoSAR Composer to flesh out the detailed behavior of each component. The integration of Simulink allowed for in-depth simulation and verification of critical requirements, ensuring each component functioned within the defined parameters.

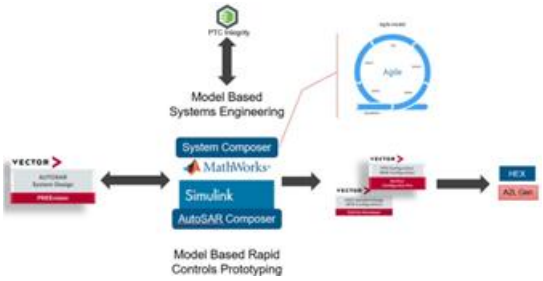

Figure 2: Enhanced Workflow with MBSE

4) Simulation, Testing, and Validation

Following the architectural design, we integrated Mathworks' Simscape [9] based Battery plant model to conductrigorous closed-loop simulations. These simulations were instrumental in verifying the majority of the BMS functionalities under various operational scenarios, mimicking real-world conditions as closely as possible.
Leveraging the powerful features of Simulink Requirements and Test Manager, we generated comprehensive test coverage and requirements traceability reports. These documents were essential for validating the simulation results and ensuring all system requirements were met. Satisfied with the outcomes from the simulation-based tests, we proceeded to utilize Simulink's automatic code generation capabilities. The code generated was then seamlessly integrated into our hardware prototyping platform, replicating the processes described in previous research papers [7].

The integration of System Composer into our Simulink-

based RCP workflow marked a significant advancement in our approach to system engineering. By enabling a more interconnected and analytically robust development process, we were able to address the multifaceted challenges of modern control systems more effectively. This strategic enhancement not only emphasized Agile development in our prototyping phase but also increased the accuracy and reliability of our system designs, paving the way for more innovative and robustproduct developments.

4. Challenges and Future Directions

As the implementation of Model-Based Systems Engineering (MBSE) and Rapid Control Prototyping (RCP) using Simulink/System Composer progresses, several challenges have surfaced that necessitate ongoing attention and resolution. Additionally, the rapid evolution of technology and growing system complexity forecast an array of future directions worth exploring to enhance these methodologies. This section discusses the current challenges and outlines potential future directions to improve MBSE and MBRCD.

4.1 Challenges

1) Integration with Legacy Systems

- One of the most significant challenges in the adoption of MBSE and RCP using Simulink/System Composer is the integration with existing legacy systems. Many organizations operate on older platforms and systems that are not readily compatible with newer model-based approaches. Bridging this gap requires:
- Development of intermediary tools that can translate between legacy workflow and new model-based workflow.
- Customization and extension of existing Simulink capabilities to better mesh with older technologies.
- Training and transition strategies to help teams adapt to new
- workflows without disrupting ongoing projects.
- MATLAB version upgrade strategy that supports seamlessmigration to newer versions of MATLAB.

2) Scalability and Complexity Management

- As systems grow in complexity, scaling MBSE and RCP methodologies to manage and analyze these large-scale systemsbecomes increasingly difficult. Issues include:
- Handling high-volume data from simulations, which can beintensive in terms of computational resources and data management.
- Maintaining model integrity and performance in the face of
- increasing complexity and interdependencies among system components.
- Developing more robust tool that can efficiently handle the complexity without compromising on speed or accuracy.
- Dynamic adjustments and optimizations in control strategies based on real-time operational data.
- Integration of field data to refine models and simulations.

4.2 Future Directions

1) Advanced Simulation Techniques

- To address the limitations in current simulation capabilities,future developments should focus on:
- Hybrid simulation environments that blend physical testing
- with virtual simulations to provide more accurate and comprehensive analyses.

2) Utilization of AI and machine learning to enhance simulation processes, enabling predictive modeling andanomaly detection within complex systems.

3) Enhanced Collaborative Tools

- Enhancing collaboration across different teams and disciplines is crucial for the successful implementation of MBSE and RCP. Future tools should:
- Facilitate better integration across various engineering domains (mechanical, electrical, software) to ensure seamless interaction and data exchange.
- Support collaboration capabilities, allowing teams to workeffectively on the same projects.

4) Adoption of Standards and Frameworks

- The development and adoption of universal standards and frameworks for MBSE, like SysML [10] can:
- Ensure consistency and interoperability between tools and methodologies used by different organizations.
- Accelerate the adoption of best practices across industries, leading to more efficient system development processes.

5. Conclusion

Model-Based Systems Engineering and Model-Based Rapid Control Prototyping represent transformative methodologies in systems engineering. With tools like Simulink and System Composer, engineers can design, simulate, and validate complex systems and control strategies more efficiently and effectively. While challenges remain, particularly in integrating with legacy systems and managing complex, large-scale projects, the potential for further advancements and broader adoption continues to grow. Future efforts should focus on overcoming these challenges to fully leverage the benefits of MBSE and rapid control development.

References

[1] J. Estefan, "Survey of model-based systems engineering (MBSE) methodologies.," INCOSE MBSE Initiative, 2008.
[2] The Mathworks, Inc., "Rapid Control Prototyping with Simulink Real-Time," [Online]. Available: https://www.mathworks.com/videos/rapid-control-prototyping-with-simulink-real-time-1623834749134.html?s_tid=srchtitle_videos_main_1_rapid%2520control%2520p rototyping.
[3] The Mathworks, Inc., "System Composer: Design and analyze system and software architectures," The Mathworks, Inc., [Online]. Available: https://www.mathworks.com/solutions/model-based-

systems-engineering.html.
[4] The Mathworks, Inc., "System Engineering: From Requirements to Architecture to Simulation," The Mathworks, Inc., [Online]. Available: https://www.mathworks.com/campaigns/offers/model-based-system- engineering.html.
[5] Object Management Group®, "REQUIREMENTS INTERCHANGE FORMAT™ (REQIF™)," [Online]. Available: https://www.omg.org/reqif/.
[6] The Mathworks, Inc., "MATLAB, Simulink, and System Composer for Model- Based Systems Engineering," The Mathworks, Inc., [Online]. Available: https://www.mathworks.com/solutions/model-based-systems-engineering.html.
[7] R. Ingole and B. Eckhart, "Hardware Prototyping using FreeRTOS while developing AutoSAR compliant application software using Simulink™," 2022.
[8] The Mathworks, Inc., "Bridging Model-Based Systems Engineering and Model- Based Design," The Mathworks, Inc., [Online]. Available: https://www.mathworks.com/videos/bridging-model-based-systems-engineering- and-model-based-design-1634884070332.html.
[9] The Mathworks, Inc., "Simscape," [Online]. Available: https://www.mathworks.com/products/simscape.html.
[10] Object Management Group®, "SYSML V2," [Online]. Available: https://www.omgsysml.org/SysML-2.htm.