Requirements-based, early stage Architecture Performance Validation on a Brake System Use Case

Sinyoung Kang[1] Marcel Gottschall[2] Sunghyun Cho[1] Torsten Blochwitz[2]

[1]Hyundai Motor Company (HMC), Republic of Korea {Newyoung,hyuni}@hyundai.com
[2]ESI Group, Germany, firstname.lastname@esi-group.com

Abstract

Automotive industry OEMs and suppliers are progressing their engineering processes and performance to the next maturity level gearing to digital thread solutions. Current challenges like continuous engineering, virtual certification, distributed development, consolidated virtual proving grounds, homologation, digital twin and operational applications, require well informed decision making in a comprehensive, reliable, traceable and customizable environment. In particular, in automotive domain, with widespread tight collaborative ecosystems between integrators and suppliers, the capability of tracing each decision and its underlying artifacts becomes a key value of an engineering platform.

This paper will outline a middleware approach to reuse generated artifacts and their relationships in a federated engineering environment and applying Modelica simulation models as the key integrator between architectural decision making and subsequent development up to detailed 3D design. Based on an exemplary, automotive brake system setup, the benefits of integrated data and workflows from specification to virtual architecture exploration and design verification are highlighted to motivate their value towards a realization of continuous model-based systems engineering methodologies.

Keywords: MBSE, System Simulation, Virtual Validation, Automotive Brake System

1 Introduction

The increasing complexity of vehicle development has led to a continuous effort to adopt system engineering-based approaches. Organic integration and management through system engineering are essential to achieve optimal performance and reliability. Recently, model-based systems engineering (MBSE) methodologies have emerged as an alternative to document-centric systems engineering, enabling a holistic view and systematic management of the growing complexity of requirements.

The vehicle architecture validation process for braking performance involves various aspects and engineering disciplines, such as braking distance, deceleration, and thermal capacity. However, past performance validation efforts have lacked systematic systems engineering review, and requirements and test case management have not been well-structured, leading to significant challenges, both in traceability and integrated performance assessment, particularly in the early stages of vehicle development. Specifically, a structured approach to linking requirements and 1D simulation models has been lacking, and MBSE methods have been proposed as a solution to systematically manage these relationships in theory.

In the daily engineering reality, the disconnection between requirements, MBSE-based architecture models, and 1D models, each managed in separate specialized software, has hindered effective traceability management.

Therefore, the primary objective of this paper is to implement a *digital thread* that organically connects and manages requirements, architecture models, and 1D performance models, covering the early stages of development of cyberphysical systems[1]. By establishing a database-centric digital thread, the aim is to build an integrated and efficient management approach that ensures traceability and consistency. Specifically, this study will leverage MBSE methodologies to effectively integrate SysML-based architecture with 1D models and design requirements. This approach not only addresses the current inefficiencies, but also lays the foundation for a more robust and scalable systems engineering process.

2 Motivation and Implementation Approach

Achieving digital continuity along the engineering cycle as shown in Figure 1, leveraging the available information and digital assets, is a major contributor to more efficient development processes forced by cost and cycle time reduction needs. However, this is not yet state of the art, in particular in real-world, multipartner, industrial and productive applications and ecosystems.

2.1 The Need for MBSE at HMC

Traditionally, product development has been conducted independently in various isolated disciplines such as mechanical, electrical/electronic, and control engineering. This approach has led to challenges in addressing issues that arise during the development process, which can result in difficulties in design, product integration, and ver-

[1]technical systems that consist of a physical part and a software/controller component

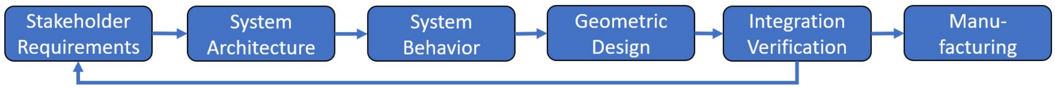

Figure 1. Flat representation of the well known V-cycle model (only major steps, excluding operation of system of interest)

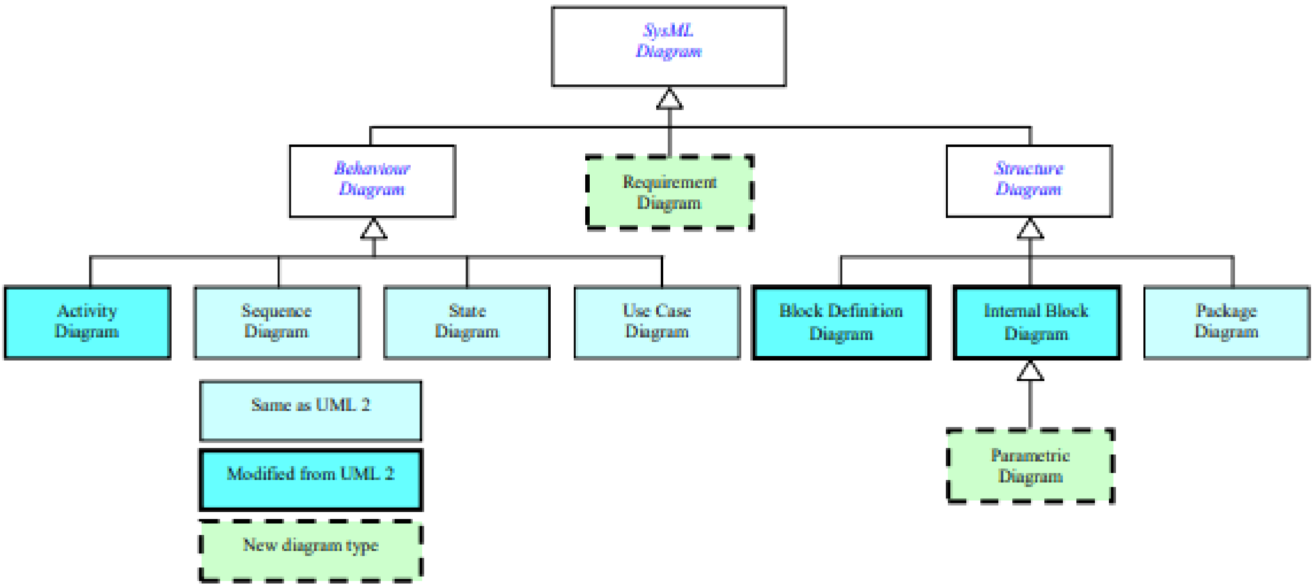

Figure 2. OMG SysML Diagram Taxonomy (Hause et al. 2006)

ification, ultimately leading to delays in product launch. With the increasing complexity and intelligence of modern products, there is a growing need to address these limitations. Model-based technologies, like MBSE have emerged as an alternative to overcome the limitations of traditional systems engineering and development frameworks. Those approaches are lacking of effective communication and collaboration across different engineering domains. These siloed methods often result in misaligned objectives, conflicting requirements, and a fragmented understanding of how changes in one area can impact others. Consequently, critical design decisions may be made without fully considering their implications on the overall system performance, leading to inefficiencies and increased risk of failure in the final product.

Hence, in this research and proof of concept application, the focus is on utilizing the MBSE methodology in the domain of brake system performance, with the aim of not only validating the approach itself within a single performance area and system, but also exploring the potential for understanding the relationships and traceability between multiple performance aspects and across multiple systems within complex, multidimensional design processes.

2.2 Architecture System Modeling

The Systems Modeling Language (SysML) is a standardized language developed to effectively implement MBSE, (Object Management Group 2022). SysML, defined by the Object Management Group, is an extensible, graphical language that can comprehensively represent the *structure*, *behavior*, *requirements*, and *parameters* of complex systems in a single model representation.

While the widely used Unified Modeling Language (UML) has been primarily focused on software-centric modeling, SysML extends it to support modeling across various domains, including hardware, software, information, and processes, encompassing the entire system. Thus it ideally covers the scope of cyberphysical systems, like the mentioned *brake system*. The key SysML diagrams shown in Figure 2 include Requirement Diagrams, Structure Diagrams, Behavior Diagrams, and Parametric Diagrams, which enable clear visualization and analysis of different aspects of the system.

Specifically, when modeling dedicated performance domains, such as braking performance, SysML offers the following advantages (Hause et al. 2006):

- **Requirement Traceability**: The Requirement Diagram (REQ) allows for the clear definition of system requirements and enables tracing them throughout the design and verification process to ensure requirement fulfillment.

- **Structural Analysis**: The Block Definition Diagram (BDD) and Internal Block Diagram (IBD) can be used to explicitly express the functional and logical structure of the system, helping to clarify the relationships between system components and maintain consistency from the design stage to the integration and verification stages.

DOI
10.3384/ecp21793

- **Behavioral Modeling**: Diagrams such as Activity Diagrams (AD) and State Machine (SM) Diagrams can be used to model the dynamic, but abstract, non-physical behavior of the system, enabling the simulation of operational scenarios and the identification of potential issues in advance.

- **Parametric Analysis**: Parametric Diagrams can be utilized to define the performance parameters of the system and analyze the system's performance under various conditions.

Based on these SysML features and capabilities, the present work will explore the benefits of such hierarchically concatenated artifacts, when they are reused in physical simulation domain. It is expected, that a *seamlessly* integrated workflow consisting of a) summarizing the verification requirements for braking performance in vehicle development, b) representing traceability of these requirements through the relationships between Requirement Diagrams and Block Definition Diagrams, and c) extracting structural analysis of the brake system architecture through Block Definition Diagrams and Internal Block Diagrams will demonstrate the effectiveness of MBSE methodology for real, cross-domain early stage engineering processes for hierarchical multiphysical systems.

2.3 Limitations of existing Approach at HMC

In current systems engineering practices, requirements are typically managed using documents such as Excel or Word, or requirement management software like IBM *DOORS*. However, it is not straightforward to link the requirements managed or handled in these documents or software to the Requirement Diagram in a SysML-based architecture model, particularly in typical multi-vendor engineering tool landscapes that are in place at enterprise level. While some SysML authoring tools like Dassault Systems *CAMEO Systems Modeler* provide plugins, these often expect the requirements to be exchanged as a ReqIF standard XML file, necessitating additional, manual steps to directly integrate the requirements in the model and continuous process.

Furthermore, there are even greater challenges in connecting the SysML architecture model to the analysis models used for actual verification. To validate the content of the SysML model, 1D and 3D analyses, or physical testing must be performed. Even if the SysML architecture model is used as a basis, significant time and effort are still required to set up the analysis models, especially for 1D, physical analysis approaches like Modelica-based solutions (Modelica Association 2021), where a separate 1D model needs to be created, despite the similarities in the structure. Recent standards, like Modelica SSP[2] (Modelica Association 2022) are tackling some of these integration aspects between architecture and simulation domains, but are still far away from productive application and dissemination.

As described above, while the benefits of MBSE-based architecture models are recognized, it remains difficult to maintain traceability between requirements, test cases, and 1D simulation models, as these are often managed in separate tools. Additionally, it is challenging to verify 1D model results against the requirements-based validation.

2.4 Digital Thread Implementation

To address the inconvenience caused by the incompatibilities in the working environments for requirements, architecture models, and 1D models, we conceived the idea of integrating the *artifacts*, created by the engineering clients, instead of connecting relevant software tools one-to-one by import/export actions.

In order to achieve such tool agnostic, vendor neutral continuous engineering platform, a microservice-based middleware approach is implemented as shown in Figure 3. Consisting of OSLC[3] compliant, domain-specific webservices, e.g. requirements- or testmanagement, this layer provides full flexibility and scalability with respect to the considered process. It stays intact even when frontend (engineering clients) or backend (data management) tools are replaced over time, thus keeping consistency throughout the linked engineering data, (Open Services Project 2021). Moreover, depending on the features of the applied backend systems, the middleware layer can also provide general support functions like user management or versioning. However, usually such capabilities are integrated in modern datamanagement backbones, e.g. PLM[4] systems. On the other side, the engineering clients are accessing the engineering data by dedicated plugins (as the only component to be adapted in case of changes in tooling), which replicate the corresponding, desired engineering workflows using the linked data that is exposed by the middleware. Hence, from user perspective, this reflects the *single source of truth* paradigm, as each participant that is part of such integrated process consumes or pushes his information from or to the same entry point without data replication.

With this technology, we connected a representative multi-vendor engineering environment, covering early stage design and verification phases:

- the requirements management software
  IBM DOORS (classic),

- the SysML architecture modeling tool
  Sparx Systems Enterprise Architect,

- and the Modelica-based 1D analysis tool
  ESI SimulationX,

to ensure traceability and enable efficient architecture-level braking performance validation. Furthermore, we

[2] System, Structure and Parametrization

[3] Open Services for Lifecycle Collaboration

[4] Product Lifecycle Management

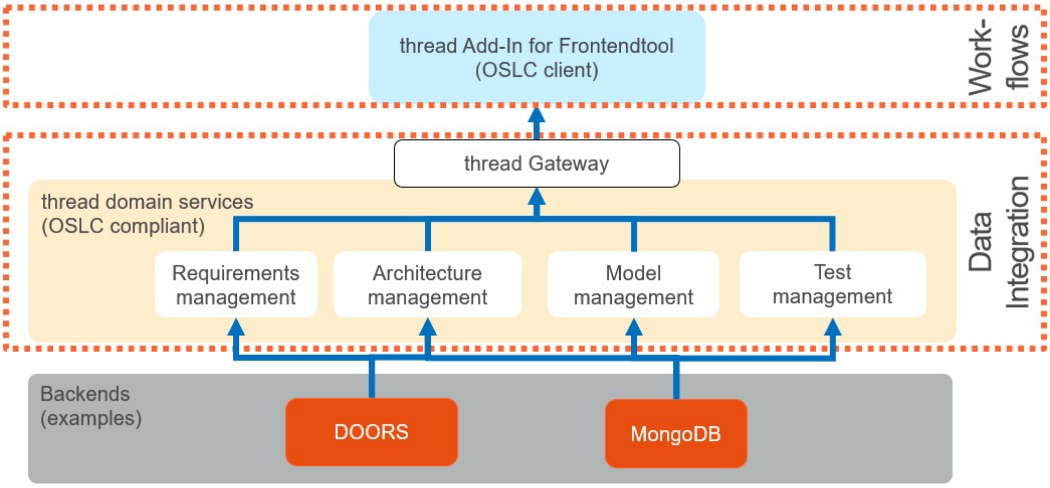

Figure 3. Tool agnostic, digital thread middleware approach based on OSLC compliant webservices

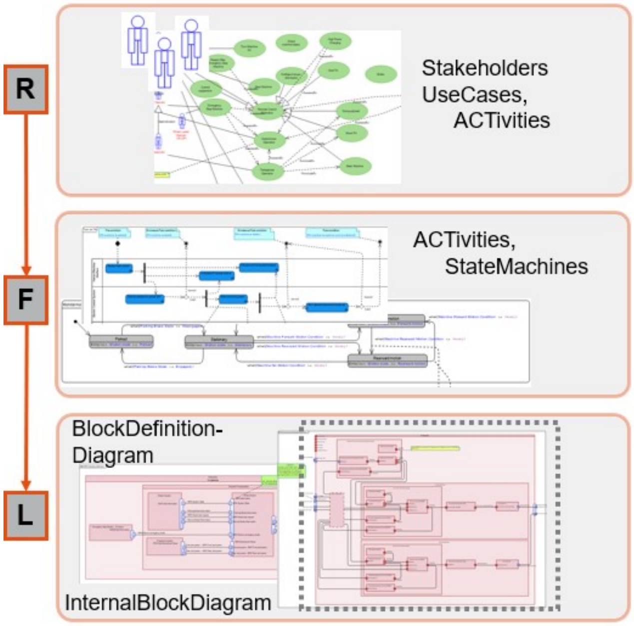

Figure 4. MBSE approach linking design artifacts, logical structure (L) *implements* functions (F) that *satisfy* requirements (R)

implemented a bidirectional traceability structure, not just a unidirectional link from requirements to the 1D simulation model. By leveraging the OSLC technology and a database-driven approach, we were able to establish a connection that allows the verification results from the 1D performance model to be reflected back in the requirements definition and management system. The application of the implementation will be demonstrated and discussed in the next sections.

3 Design Methodology

Following the principles of MBSE, a formal and clearly structured process, shown in Figure 4, enables the derivation of structural descriptions (components and their interrelation) of the target architecture, as the main input for the simulation domain. Starting with high level requirements or stakeholder needs (R) on the system of interest, a functional architecture (F) is designed which satisfies the requirements. Applying various abstract modeling and analysis descriptions like Use Case- and Activity Diagrams or State Machine simulations, enable the (non physical) justification of the design and a further decomposition of the functional behavior into logical components (L). Apart from logical and discrete simulations, the systems architecture has to be verified against performance requirements by 1D physical simulation, before propagating system sizing and design parameters to the 3D design stage.

However, historically, systems engineering (holistic) and simulation (domain experts) have been disconnected, which results in a gap in the level of detail for the logical structure description (L), as the architecture definition in MBSE output is still abstract and does not fulfil physical simulation needs. This issue, recently discussed in (Cederbladh et al. 2024), has to be handled and adressed, when aiming for a cross-domain digital thread implementation with seamless integration from architecture to performance simulation. Applying stereotyping to connections between different logical components (L), maps the abstract layer of description in SysML language (e.g. "two components are exchanging energy") to the information in physical domains (e.g. "electrical connection is required between motor and battery"), that is necessary to build corresponding physical simulation models to be used for the architecture verification. Hence, these physical models are not only serving the actual performance validation activities, but also enable the detailing and specification of interface definitions on the logical layer based on simulation justification, which is absolutely required for complex systems, early stage architecture exploration activities. With this extension to SysML model descriptions, it becomes possible to automate the reuse of architecture information for generating physical simulation model tem-

DOI
10.3384/ecp21793

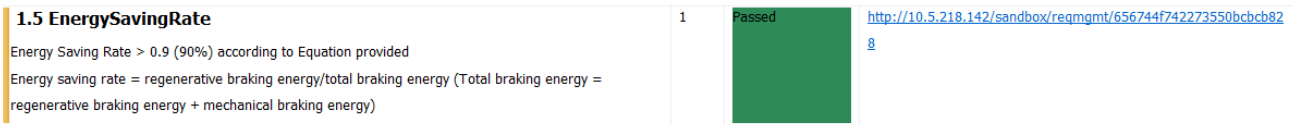

Figure 5. Specific case for brake thermal management, representation of the performance requirement in DOORS with content (first column), version (second column), virtual verification result (third column) and ID of OSLC artifact (fourth column)

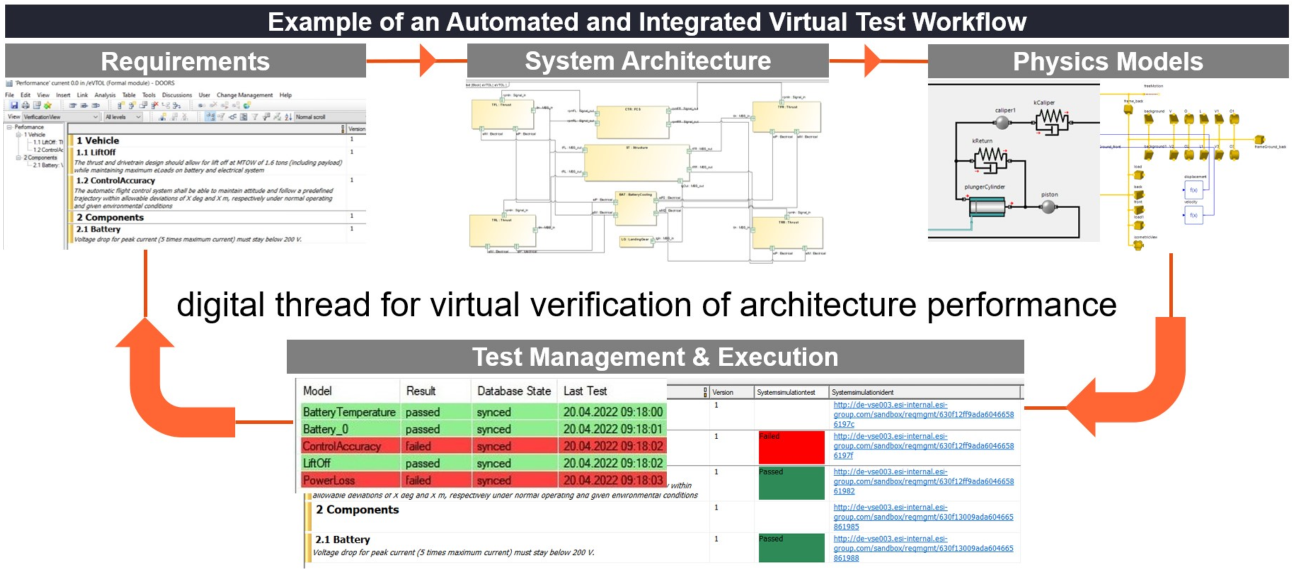

Figure 6. Closed loop, MBSE-based early stage, architecture and design verification using 1D Modelica models replicating virtual test scenarios defined by system requirements (REMARK: due to IP restrictions, the screenshots do not show the actual brake system models, but demonstrate the concept and workflow)

plates, that are linked to certain requirements and used in subsequent performance analysis tasks. Another benefit of this cross-domain integration from design perspective, is the inheritance of global architecture design parameters that automatically become (Modelica) parameters of the resulting behavioral models, as demonstrated below.

3.1 Brake System Performance Use Case

To validate the braking performance, the vehicle development requirements managed internally by the company were consolidated, serving as a foundation for establishing the objectives to be verified through a 1D simulation model. It should be remarked, that there are of course plenty of structural and performance requirements on the brake system, referring to different hierarchy levels (subsystems, components), which are qualified for virtual verification by simulation and have been used in this study, e.g stopping distance with ABS or several system- and unit fail scenarios. However, to demonstrate an efficient, integrated validation process, the *Energy Saving Rate* verification shown in Figure 5 is selected as a representative case, as the Modelica-based multiphysics model includes brake hydraulic, mechanical, thermal and electrical system behavior for simulating braking performance .

The Energy Saving Rate (ESR) is obtained through energy analysis. First, energy is calculated by integrating power during braking periods in driving cycles, such as the WLTP[5] for Hybrid Electric Vehicles (HEV) and Electric Vehicles (EV) that are equipped with regenerative braking. Then, the calculation ESR formula is as follows:

- ESR = Regenerative Braking Energy / Total Braking Energy, with
- Total Braking Energy = Regenerative Braking Energy + Mechanical Braking Energy

Each company has established target values for the ESR, which is considered confidential information within each organization. Therefore, a sample value of approximately 0.9 has been set for the target in this study.

3.2 Process Demonstration

As described above and summarized in Figure 6, the virtual validation process consists of 4 major steps, seamlessly linking and integrating the disciplines of requirements, functional and logical architecture and physical simulation models for design and verification tasks. With the implemented digital thread, a closed-loop workflow with digital continuity is established, enabling collaboration between the R-F-L-P and test domains. Such setting will allow for certification credits based on simulation results, as end-to-end traceability is one of the key require-

[5] Worldwide Harmonized Light Vehicles Test Procedure

ments in regulations like ISO 26262, (Peraldi-Frati and Albinet 2010; Maro, Staron, and Steghöfer 2017).

The braking system and performance requirements established earlier, are stored in the digital thread database backend by a plugin in the DOORS client. Please note, that this digital thread backend is running in parallel to databases that are potentially in place when common engineering tools client/server architectures are deployed. Depending on their flexibility with respect to the datamodel modifications and extensions, another multipurpose database might be required to handle the linkages between the different engineering artifacts. In a second step, the requirements are retrieved into the architectural design tool Enterprise Architect, again using a dedicated plugin. This way, the requirements and their hierarchy are automatically added to the SysML model, as a starting point for the functional design step.

Using the requirements retrieved via the digital thread, the braking functional elements were created in the architecture model, represented through requirements diagrams and block definition diagrams. Additionally, the physical structure necessary to implement the defined functions for brake validation is depicted in an internal block diagram, facilitating architecture modeling based on physical structure analysis. This process allows for the definition of traceability among requirements, functional definitions, and physical structures, with their interrelationships expressed through connections between blocks. A more detailed description of the different, formalized steps can be found in Aleksandraviciene and Morkevicius (2018). It is one of the key strengths of a holistic systems model representation, to easily express the allocation of architectural components to certain requirements for verification purposes in a user convenient, graphical way.

After the logical structure breakdown is derived from the functional design, the various connections between the different components are allocated to dedicated physical domains, like hydraulics, electrics or mechanics, using stereotyping[6]. This step drives an active collaboration and iteration between the systems- and simulation engineer, as the latter is the domain expert on specific systems and has knowledge on potential limitations or boundary conditions when mapping physical realizations to a certain logical structure. In addition, the already mentioned architecture design parameters are defined and added to the corresponding logical components. Such parameters refer to system level, *global sizing* or *structural variants inside a certain architecture* descriptions and are reused to define simulation runs downstream in the verification process.

The physical structure - or more precisely, the logical structure augmented with physical information - represented in SysML IBDs, is organized specifically for the brake system to evaluate braking performance. In particular, for complex systems with many hierarchical levels, the corresponding physical models become big with high computational efforts, leading to weak performance in use cases like architecture exploration with large amount of simulation runs. Hence, the concept of *partial IBDs* is introduced for improved virtual testing performance. As shown in Figure 7, dedicated IBDs, focussing on specific aspects of a complex logical component, like thermal or mechanical domain, are extracted from the inner structure described by the full (multiphysics) architecture IBD. These partial IBDs only show components and subsystems, that are contributing to the specific aspect of the component or system, like the thermal behavior in the current demonstration use case. As mentioned above, these architectural components are then linked or allocated to the corresponding (sub-system) requirements. Hence, there is a clever separation between the reduced architectural models used for physical verification runs (here: the thermal management model), and the full logical structure representation (here: the brake system model), used for design descriptions like *bill of materials* of a system. Another benefit of this approach is the easy support of function-based design and testing methods in the 1D simulation domain, because of the allocation of the reduced architectural models to functions and requirements, following the R-F-L process. In summary, structural information in (partial) IBDs and linking of architecture to requirements are key information and input for the subsequent step of system simulation in the design process. The available, integrated information of a system adheres to the Model-Based Systems Engineering methodology, enabling the direct reuse of the internal block diagram for cross-domain linking of architecture- with the corresponding 1D simulation model.

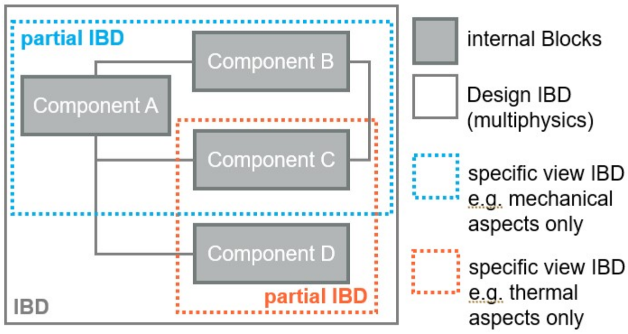

Figure 7. Concept of partial IBDs as representation of specific aspects, e.g. a single physical domain, of a logical component for integrated virtual verification purposes

Similar to the requirements, relevant information of the SysML model from process perspective, like the IBD representing the physical structure of the brake system, or tracelinks between requirements and the internal block diagram, are uploaded to the digital thread environment's database using the plugin developed for Enterprise

[6]adding additional information to modeling components

DOI
10.3384/ecp21793

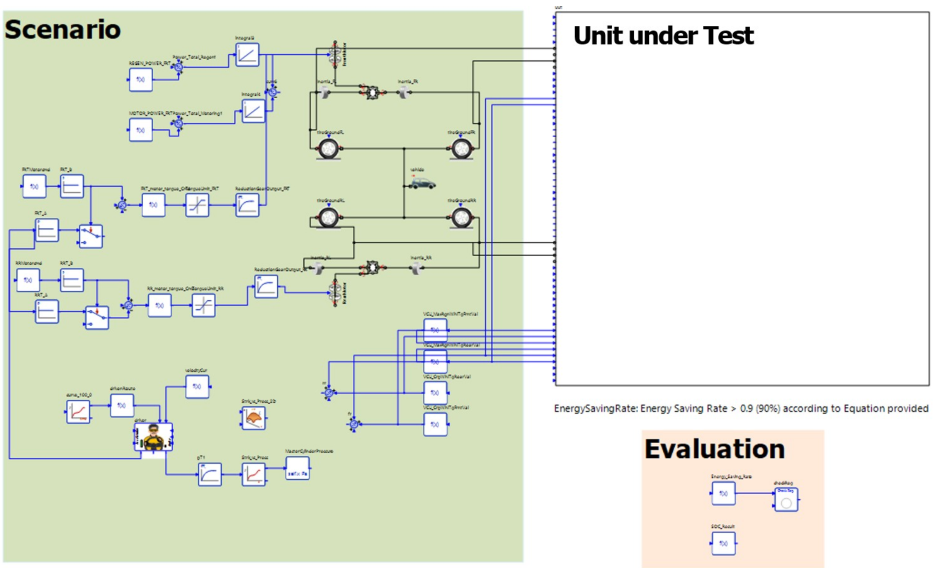

Figure 8. Testmodel (simplified) for the thermal management requirement example, consisting of 3 main components: a) design model instance (*Unit under Test*), b) simulation, environment and parameters (*Scenario*), and c) test verdict extraction (*Evaluation*)

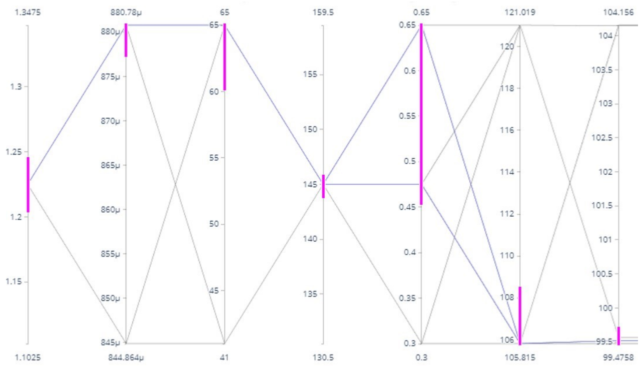

Figure 9. Design parameter study (sizing) to identify simulation runs within a desired range of model parameters (pink bars) using python scripting capabilities integrated in the model

Architect. This is required, as the version of the tool used in this study is not connected to any data backend but relies on local file storage.

In the following, by utilizing the SimulationX plugin, a connection is established with the database through the digital thread, enabling the automatic generation of Modelica libraries (packages) and types (models), that correspond to and reflect the structure of the IBD of the SysML architecture model. These automatically generated Modelica templates implement the connections and ports as well as the system hierarchy represented in the IBD. However, it is important to note that the Modelica blocks derived from the SysML architecture model are not immediately executable models, they merely represent the architectural structure of the brake system. Therefore, to conduct a 1D performance analysis, it is necessary to add components capable of performing such operations.

Thus, the simulation engineering process is taking place in a next step, creating physical models inside the templates, describing the actual behavior of the components. Please be aware, that the model generation can only be automated based on available information, and as mentioned above, SysML language is limited to an abstract layer of system modeling not intended for physical simulation. This physical modeling can happen, either by using native Modelica models from the library, e.g. a valve or heat exchanger, or reusing models from other, external sources by means of FMI, (Modelica Association 2014), e.g. a controller modelled in The Mathworks *Simulink*. The internal components of the brake system Modelica blocks are detailed to enable 1D analysis of braking performance. Again, the engineering artifacts (system simulation models) are shared with the stakeholders by uploading the information to the digital thread database using the SimulationX plugin.

To enable the actual virtual verification and to measure brake performance, the physical models of the brake system architecture have to be augmented as shown in Figure 8. Now, the previously created tracelink between architectural components and requirements are reused by the SimulationX plugin to support the simulation engineer and show a list of *test cases*, consisting of requirements that need to be tested by simulation and are already allocated to structural components. By selecting a certain test case, a *testmodel* is automatically generated, instantiating the appropriate Modelica component from the generated system library as *Unit under Test* in the figure above. The architecture design parameters, discussed before, are present on the corresponding Modelica component and inside the Unit under Test block, and can be used to execute variants studies (design space or architecture exploration) or component optimization tasks as shown in Figure 9 within a given system structure. Such analysis

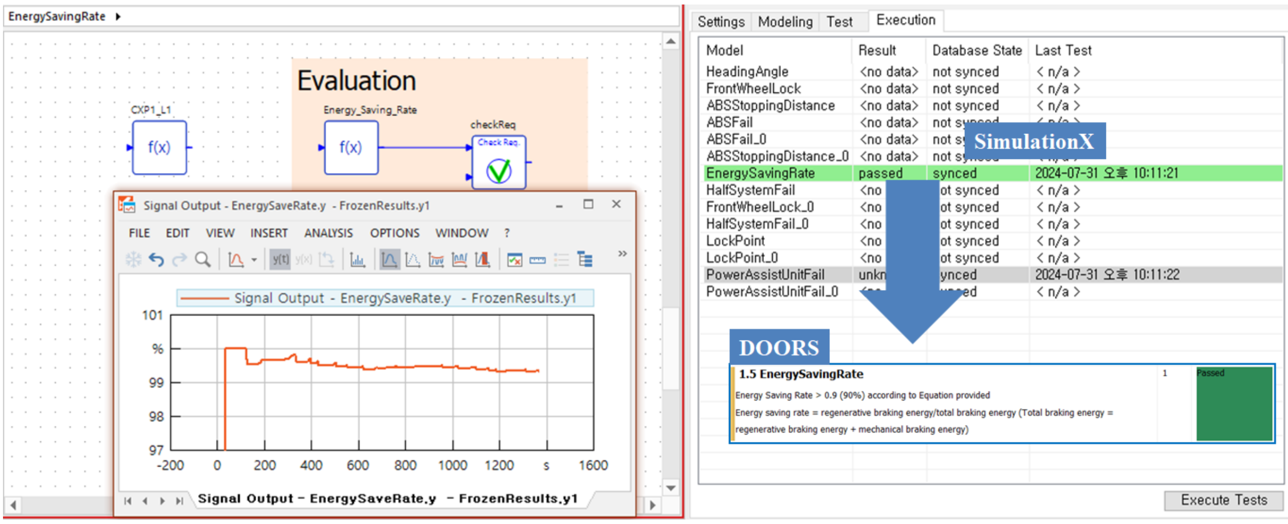

Figure 10. Evaluation result of braking scenario for thermal management requirement use case, transient behavior in the Modelica model (left), test verdict representation in tool plugins (right)

workflows can be automated easily by considering only valid simulation runs, since the model is connected by the digital thread to requirements to be tested and fulfilled.

In order to replicate a test scenario described by the linked requirement, additional components need to be modeled by the simulation engineer. As shown in the thermal management model above, environment and stimulation incorporating virtual vehicle models, including the driver, signals, vehicle weight, and wheelbase, etc. are added to the model collected in the green *Scenario* part. Moreover, the testmodel is augmented by the test verdict analysis, using dedicated feature extraction and requirements fulfilment elements[7] from the associated SimulationX Modelica library, collected in the orange *Evaluation* part. Finally, these ready to use simulation models are synchronized with the digital thread database using the SimulationX plugin. With the defined and established links along the process and end to end traceability from requirement to virtual verification (simulation) result is deployed.

In a last step closing the loop shown in Figure 6, the test results are synchronized with the requirements management system DOORS using the corresponding plugin as shown in Figure 5, thus realizing a (simple) test management and tracking system. Figure 10 illustrates the verification of the ESR using the discussed thermal management 1D simulation model and demonstrates, how traceability between the verification results of the braking performance 1D model and the requirements is established in this proof of concept study.

4 Discussion of Decision Making

The demonstrated requirements-based early stage architecture performance validation process is expected to bring significant benefits to daily engineering and design efforts. Developing and certifying commercial vehicle components is challenging, in particular in the demonstrated example of safety critical elements like braking systems. Because of the complicated interrelations between the various systems, sub-systems and components, new techniques for digital continuity, reusability and traceability of development artifacts need to be applied, to handle the complexity throughout the development cycle. With the MBSE methodology, a holistic view and tracking along the increasing maturity level of the system of interest becomes possible in real life applications. Based on the natural, structured R-F-L-P steps, a complex cyberphysical system with multiple hierarchy levels and tight interactions with other systems, gets feasible to manage. Applying simulation results from multilevel models (from simple 0D to 1D/3D) for integrated decision making gives confidence on every decision gate from one domain to the other, throughout the early stage architecture design and verification process. This incremental approach allows to cascade and derive lower level, more detailed requirements on sub-systems and components based on the initial, top level stakeholder needs on product level.

However, it should be noted the introduction and deployment of this process will have a considerable impact on current workflows and will necessitate changes in both processes and personnel (mindset). Apart from these general side effects, the key attributes and considerations from industrial users perspective are summarized below as they share positive impact on engineering and design processes, contributing to a more efficient and systematic working environment.

Benefits in Daily Engineering and Design Tasks: By preventing redundancy and omissions in requirements and functions while ensuring traceability, engineers will have clearer criteria for their work. This will reduce ambiguity in communication and facilitate collaboration among team members.

[7] providing a "passed", "failed" or "undefined" result depending on specified conditions

DOI
10.3384/ecp21793

Modeling Guidance: Guidelines have been developed and need to be applied to map SysML models to simulation requirements. As the SysML language is "method agnostic", the degree of freedom for the systems engineer creating the logical structure needs to be limited to enabled the seamless reuse of information in the subsequent, physical model-based design. These guidelines enhance the consistency of modeling efforts for architecture design and performance validation, supporting effective decision making.

Complexity Resolution and Consistency in Decision Making: By structuring complex and diverse requirements to build the system architecture, it becomes possible to analyze the system before detailed design. This approach allows for easy tracking of components affected by changes and provides linkage information for impact analysis. Furthermore, decision making based on structured information and relationships during the MBSE process supports objective and consistent decisions, independent of the decision maker's intent or the engineer's expertise.

Role of Digital Thread: The digital thread plays a crucial role in systematically connecting information through model-based approaches, enhancing the overall understanding of automotive systems through integrated analysis. Furthermore, when multiple personnel can collaborate via a database-driven digital thread, it facilitates more efficient teamwork and information sharing.

5 Conclusions

This study proposes a Model-Based Systems Engineering (MBSE) methodology, extended to and integrated with model-based design activities, as a means to address the complexities involved in vehicle design and validation, specifically within the context of performance validation for braking systems at early stages of vehicle development. The implementation of digital threads has facilitated the integration of requirements, abstract architecture models, and 1D physical simulation models. The significance of this research and proof of concept application, from an industrial early adopter perspective, can be defined as follows:

- The methodology provides an integrated and visual modeling environment for vehicle development researchers, demonstrating how MBSE-based, design and validation processes enhance quality and reliability.

- By connecting requirements, architecture-, and 1D simulation models through cross-domain digital threads, the methodology enables more efficient workflows. Notably, the internal block diagrams of SysML architecture models can be directly transformed into Modelica blocks as components of 1D models, thereby reducing the inconvenience of reworking defined elements from SysML in the performance model, thus improving overall process efficiency.

- The verification results of requirements achieved with the 1D simulation models can be directly validated and documented in the requirements management software, establishing a closed-loop requirements-based verification framework.

However, the applied MBSE methodology is currently implemented and demonstrated on the validation of braking performance at the early stage of vehicle development, and the integration with other performance domains and software remains a significant challenge. Particularly, despite being at the early stage, verification through 3D analysis presents constraints that hinder the direct reusage of SysML architecture modeling results, similar to the 1D simulation domain. Nonetheless, in other performance areas where verification is conducted using software, based on the Modelica language, it is anticipated that implementing connections through digital threads, as demonstrated in this study, will lead to more efficient operational directions. A concrete idea in this direction is the seamless reuse and continuous integration of architecture, simulation models and scenarios in the virtual verification of Electronic Control Units by SiL and HiL[8] testing methods. Further automation of test model generation, based on available SysML descriptions like sequence- or parametric diagrams, will add on top of the aforementioned improved usability and efficiency.

Additionally, the utilization of automated parameter value selections in heterogenous verification environments (like co-simulation of Modelica and CarMaker) becomes possible and can significantly enhance development efficiency, (Lu et al. 2021). This would leverage the potential of our digital thread integration approach to streamline processes and improve outcomes in vehicle system design and validation at HMC and similar industrial applications.

References

Aleksandraviciene, Aiste and Aurelijus Morkevicius (2018). *No-Magic Magic Grid - Book of Knowledge*. 1st ed. Vitae Litera.

Cederbladh, Johan et al. (2024). “Correlating Logical and Physical Models for Early Performance Validation - An Experience Report”. In: *IEEE Systems Conference SYSCON2024*. IEEE.

Hause, Matthew et al. (2006). “The SysML modelling language”. In: *Fifteenth European systems engineering conference*. Vol. 9, pp. 1–12.

Lu, Jinzhi et al. (2021). “Model-based systems engineering toolchain for automated parameter value selection”. In: *IEEE Transactions on Systems, Man, and Cybernetics: Systems* 52.4, pp. 2333–2347.

Maro, Salome, Miroslaw Staron, and Jan-Philipp Steghöfer (2017). “Challenges of establishing traceability in the automotive domain”. In: *Software Quality. Complexity and Challenges of Software Engineering in Emerging Technologies: 9th International Conference, SWQD 2017, Vienna, Austria, January 17-20, 2017, Proceedings 9*. Springer, pp. 153–172.

[8]Software- and Hardware-in-the-Loop

Modelica Association (2014-07). *Functional Mock-up Interface for Model Exchange and Co-Simulation Version 2.0*. Tech. rep. Linköping: Modelica Association. URL: https://fmi-standard.org.

Modelica Association (2021). *Modelica – A Unified Object-Oriented Language for Systems Modeling. Language Specification Version 3.5*. Tech. rep. Linköping: Modelica Association. URL: https://specification.modelica.org/maint/3.5/MLS.html.

Modelica Association (2022). *System Structure and Parameterization Version 1*. Tech. rep. Linköping: Modelica Association. URL: https://fmi-standard.org.

Object Management Group (2022). *OMG Systems Modeling Language Version 1*. Tech. rep. Massachusetts: Object Management Group. URL: https://www.omg.org/spec/category/systems-engineering/.

Open Services Project (2021). *Open Services for Lifecycle Collaboration Version 3*. Tech. rep. Open Services Project. URL: https://open-services.net/.

Peraldi-Frati, Marie-Agnès and Arnaud Albinet (2010). "Requirement traceability in safety critical systems". In: *Proceedings of the 1st Workshop on Critical Automotive applications: Robustness & Safety*, pp. 11–14.

DOI
10.3384/ecp21793