Modelica as Model Aggregator for holistic Architecture Validation of Electric Vehicles

Marcel Gottschall[1] Torsten Blochwitz[1] Andreas Abel[1] Alex Magdanz[2]

[1]ESI Germany GmbH, ESI Group, Germany, name.surname@esi-group.com
[2]Xcelerated Prototyping Inc., Canada, alex@xpincorporated.com

Abstract

Automotive OEMs and suppliers are facing recent challenges in the development process, induced by ever shortened product cycles, further distributed development as well as increasing demands for virtual testing and certification using virtual proving grounds or digital twins.

This paper presents a real-life demonstration of a federated, seamlessly integrated design process for a complex cyberphysical system (electric truck), where simulation is used for early-stage performance validation and decision making. Since holistic, but abstract architecture models created in systems engineering discipline contain relevant information with respect to logical system structure and allocated requirements, the simulation domain will benefit from a cross domain linking of model artefacts. By aligning system interfaces across model abstractions and augmenting logical models with physical information, behavioural model templates for design can be generated in a smart, traceable and automated fashion. With the additional information of requirements allocated to certain architectural components in those abstract architecture models, it is demonstrated how scenario-based component and system simulation will contribute to analysis tasks like architecture exploration or specific design optimization in efficient, continuous engineering environments.

Keywords: Digital Thread, MBSE, Virtual Testing, Electric Vehicles Architecture

1 Introduction and Engineering Ecosystem

Ongoing digitalization of todays product lifecycle, from development to operation, creates new opportunities, but also new challenges need to be handled introduced by the ever increasing complexity of products and their underlying processes. Traditionally, product development is divided into stages with clear separation, based on each discipline's view of the specific system of interest (see Figure 1), to enable systematic processes at each design step. Systems Engineering (SE) is such major process, taking place at the architectural level of mechatronic or cyberphysical systems[1], physical performance design (1D) or geometrical specification (3D) are successors in a natural top-down workflow. As shown in Figure 1, the very first phases of a digital, holistic product definition, from stakeholder needs to system functions and logical implementation to the physical realisation, are commonly included in the term *model-based systems engineering* (MBSE).

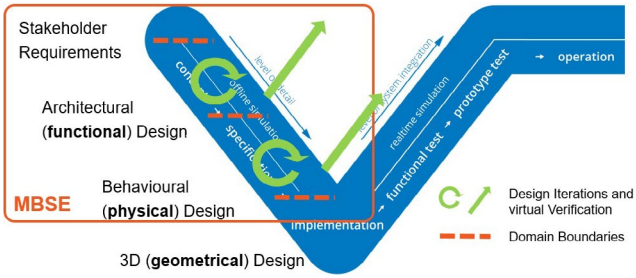

Figure 1. Generic representation of product lifecycle with focus on early-stage phases of design and the involved engineering disciplines, commonly allocated to model-based systems engineering terminology (MBSE)

Systems engineers describe the principal system architecture and its requirements to be fulfilled in early development stages with abstract models, most commonly in *systems modeling language* SysML or forks thereof, (Object Management Group 2022). Despite rare cases, these logical models are lacking real representations of the physical behavior of the system. On the other hand, those are often created in later design stages by design engineers for detailed analysis of the system use cases. The Modelica modeling language is a well established method, in particular for such analysis of the dynamic, non-linear behavior of multiphysics systems. Due to limited simulation capabilities in architecture tools (SysML), there is still no possibility to effectively and conveniently link more sophisticated physical models created in different modeling platforms by distributed engineering teams involved in the design. However, providing physical system models to systems engineers would enable complex tasks, like architectural exploration and early-stage decision making as well as virtual testing of allocated requirements. In addition to that, it is important to remark, that bottom-up processes like change management and impact analysis (e.g. in case of findings on a lower, more detailed level) will benefit, or in the first place will become possible, by sim-

[1]products or systems containing a physical part, often called *plant* and a software or logical part, often called *controller*

ulation capabilities on (early stage) holistic views of complicated and complex products like cyberphysical systems with tight interdependencies between the different components.

Hence, this paper presents a new approach by integrating physical (1D) models into an architecture representation of an electric truck, which is transferred from modeling (SysML) to simulation domain in early design stages. This way, physical models and corresponding data like parameters are linked to SysML model representations of requirements, functional and logical views, as a key aspect of traceability and certification by simulation. Based on the linked data approach, relevant information from the SysML model is automatically transferred into a corresponding Modelica model components library with all system- and subcomponents (e.g. battery, drivetrain, battery cooling system) and their interaction represented by Modelica connections. Incorporating existing 1D models available in such federated, multitool engineering environments and augmenting these architecture component models with certain stimulation and evaluation creates dedicated testmodels ready to use in virtual validation campaigns. It is emphasized, that such closed loop, fully automated testmodel execution and analysis, enables early-stage architectural and design decisions as shown in Figure 1, their monitoring and evaluation considering full variability in large product lines with respect to a complete set of requirements, hence enabling agile design changes in case of failed tests.

2 Motivation and Implementation

Looking at the very common visualisation of product lifecycle by the V-Model shown in Figure 1, explains that systems architecture design and system performance simulation are direct neighbours, where the latter is consuming major output from the previous stage and vice versa. However, these disciplines are separated by their specific workflows, tools and artefacts they deal with. Digitalisation allows to break these silos of knowledge and establish a consistent and continous information flow across the original boundaries.

From this perspective, system simulation is the perfect tool to execute verification and validation steps on architectural level, providing proof and confidence on performance to enable correct decisions in typical large architecture design spaces before going down in detail and spending effort on the next level of system description like CAD or FEM. Rework costs and time-to-market are significantly decreased by such approach, thus implementing an agile methodology, known from software design and mapped to physical systems in Figure 2.

Due to its design and because of the multiphysics system character, system simulations and in particular those implemented by Modelica language, e.g. (Modelica Association 2021), are well suitable for several validation stages along the design cycle. Starting with simplified or

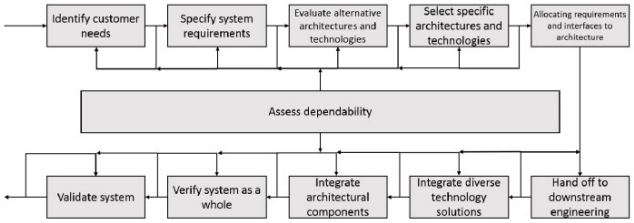

Figure 2. Detailed pocesses in early phases of product development for integration between architecting and simulation in agile systems engineering methodology, (Douglass 2016)

even surrogate model descriptions of components in very early phases of architecture drafts with lots of unknowns, down to more detailed and sophisticated models at higher maturity levels of the design throughout the system decomposition phase, system simulation perfectly serves the intended purpose. Hence, it is the key integrator between architecture and geometry of a system.

The benefits of reusing information between these domains have been already discussed and demonstrated in several ways and projects, e.g (The INTO-CPS Association 2018). Applying a predefined SysML profile in architecture models allowed the automatic preparation of co-simulation and proper parametrization between different simulation objects, thus representing a system model. However, the complexity of the models and logical structure have been quite limited, and the creation of dedicated diagrams providing the required information imposed additional modeling effort aside of the systems engineering process.

Nevertheless, to emphasize the natural combination of architecture and simulation, *Modelica Association* defined a dedicated standard called SSP[2], to apply simulation models in an architectural (= *structure*) context, (Modelica Association 2022b). Different to that approach and the available implementations, in this paper the relevant information is reused from the systems model (SysML). Such data, at first the logical structure or decomposition, meaning the different subsystems and components as well as their interrelations, enables an automatic generation of a model template and corresponding library as shown in Figure 3. This template replicates the system structure and hierarchy without any modeling effort. Hence, the design process time is significantly decreased, while it becomes more reliable regarding missmatching subsystems and components as the degree of freedom of the design engineer is limited because of predefined ports and connections. However, it should be noted, that some information have been traditionally missing in systems engineering domain (SysML) and need special treatment to bridge the gap from abstract functional system perspective to behavioural or physical view, (Cederbladh et al. 2024). Similar to SSP approach, the actual performance simulation models will be then integrated by means of

[2]System, Structure and Parametrization

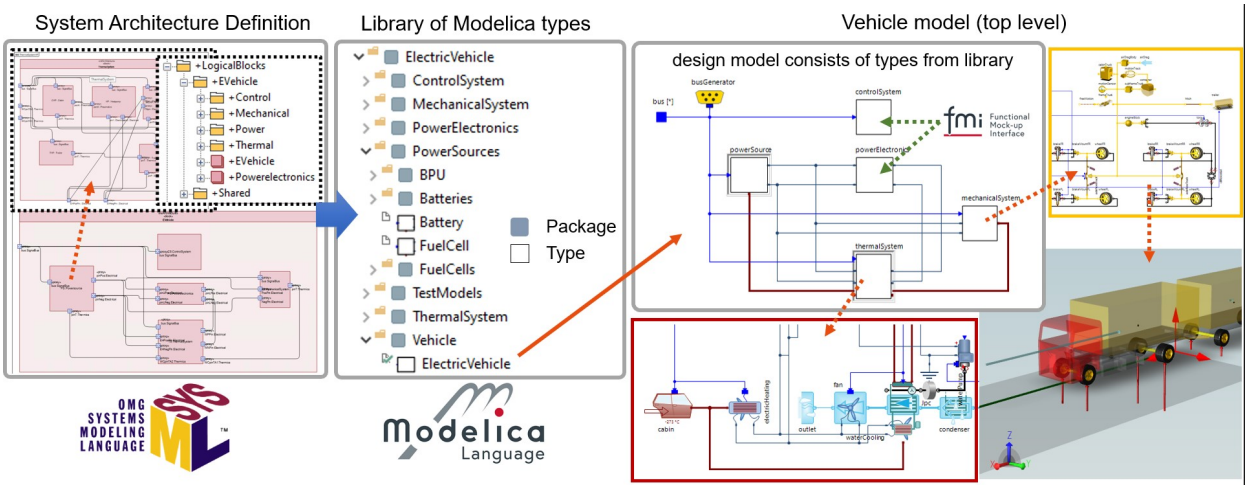

Figure 3. Cross-domain process to replicate system architecture information (logical structure) in automatically generated simulation templates to aggregate native Modelica models or existing models from other sources by means of FMI

functional mockup interface FMI or native Modelica components into these architectural templates. In particular, the latest version 3 (Modelica Association 2022a) with the support of physical connectors is a major improvement towards automation and user convenience for the aspect of continous integration between architecture and simulation, see next section for details from practical application perspective.

In a second step, the requirements or stakeholder needs, which are connected to architectural components by trace links in the SysML model, are used to extend the previous architecture simulation models for the purpose of mission- or scenario-based virtual architecture validation, as introduced above. A more detailed, walktrough visualisation of MBSE artefacts integration from requirements to virtual testing is given in (Gottschall, Binder, and Castel 2022).

At this stage in the design cycle, the application of the SSP technology mentioned above becomes obvious by exporting such full architecture simulation models in a tool agnostic model exchange container for collaborative use cases.

In order to achieve such digital cross domain integration (not limited to architecture and performance), we developed and applied a *linked data* approach, based on microservices which are compliant to open standard specification OSLC[3], (Open Services Project 2021). As shown in Figure 4, these webservices are acting as a middleware between frontend and backend tools, exposing all relevant information, collected in a multidomain datamodel of overlapping entities, and providing consistency throughout the various managed artefacts that are created by the stakeholders of the process. Engineering tools are connected by clients which implement discipline specific workflows on that data. This way, the digital thread approach becomes tool agnostic, since the frontend and backend tools can be replaced while the datalayer stays intact. Such non intrusive implementation supports the *best tool for the job* paradigm, allowing the user to keep his established ecosystem and dedicated workflows.

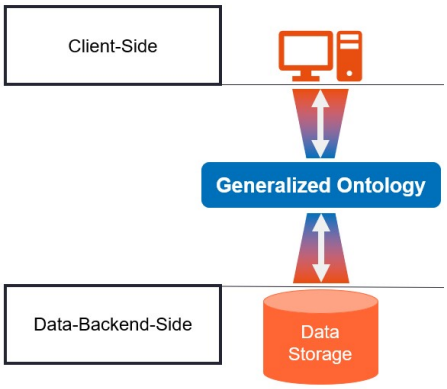

Figure 4. Middleware integration approach between engineering frontends (e.g. simulation tools) and data backends establishing coherent engineering artefacts to achieve a tool agnostic digital thread along product lifecycle with contributions of various disciplines

The benefits of continuous integration between systems engineering and simulation for early-stage development tasks will be demonstrated in the next section by top-down virtual design of a rather complex electrical vehicle (EV) system, applying incremental performance validation and corresponding decisions to highlight the value of scenario-based system simulation applications.

3 Complex System Development Process Demonstration

As shown in Figure 1, a representative development process starts with high level, mostly abstract requirements or stakeholder needs. The subsequent system description and

[3]Open Services for Lifecycle Collaboration

decomposition in the design phase for products or systems with sufficient level of complexity (like an EV) follows MBSE methodology in a top-down fashion, including the major abstraction layers and their relations:

- Requirements (R) are *satisfied* by functions (to be provided by a system)
- Functions (F) are *fulfilled* by logical components (contained in a system)
- Logical components (L) are *implemented* by physical models
- Physical models (P) are used to validate systems (subsystems, components, et.) against requirements

Such formal, hierarchical process allows breaking down a complex design task, where requirements are progressively derived and propagated along the different levels of detail or abstraction from system (e.g. electric vehicle) to subsystem (e.g. electrical system) to components (e.g. battery), always based on the simulation results and decisions made the step before, indicated by the green arrows in Figure 1.

Use Case and Sample Tooling

With the demonstration, a model-based, incremental top-down design and operation optimization use case is executed, applying either an electric truck (long distances) or electric bus (short distances, not shown) ecosystem, which results in different system architectures to be selected based on the simulation results. Here, the exemplary engineering tooling listed below is used for visualisation purposes at the different stages of the process (as mentioned above, tools can be replaced by the user):

- *PTC Windchill Modeler* (requirements and architecture modeling, SysML)
- *ESI SimulationX* (1D modeling and simulation environment based on Modelica)
- 3rd party 1D model sources like *Simulink, GTSuite, Dymola*, etc. providing FMU
- *ESI VCM*[4] (webbased virtual test management and executation environment)

Depending on the type of utility vehicle, e.g. longrange truck or city bus, the corresponding target and missions are specified and applied for the product level stakeholder needs, exemplarely listed below. Such performance requirements will be the entry point into and drive the design and verification process considering certification standards like ISO 8714, or safety regulations on components like fail safe battery design compliant to ISO 6469 or SAE J2929:

[4]Validation Campaign Manager

- **Range:** *The vehicle must be capable to serve a distance of 350 km (+50 km safety margin), or a cycle time of 4,5 hours, respectively before recharge or refill is required.*
- **EnergyConsumption:** *The specific energy consumption must be below the threshold of 1,20 kWh/km on customized, specific missions at maximum payload.*
- **ClimbingPower:** *The vehicle shall maintain a velocity of 80 km/h at a road gradient of 7 percent with maximum mass.*
- **CabinComfort:** *The thermal system must maintain a cabin temperature between 18 - 25 deg Celsius, in outside operating environmental conditions between -40 and +50 deg Celsius with respect to heating and cooling performance.*
- **BatterySafety:** *The maximum battery temperature under peak load conditions must stay below 70 deg Celsius.*

Apart from technical requirements, also economical aspects may be considered and evaluated based on the simulated loadcases (still on high, abstract level) like

- **Costs:** *The estimated, selected vehicle architecture operating costs must stay below 0.40 €/km, also considering personnel costs spent on downtime like recharge cycles.*

The payload of the EV or capacity of the power source are design parameters among others listed in Table 1, reflecting variability between different configurations of the same system architecture. It should be remarked, that the given high level requirements are cascading along the system decomposition and maturizing, where additional requirements on the sublevels, e.g. recuperation or charging power, are derived and validated accordingly.

Table 1. Examples of architecture design parameters for the EV

Subsystem	*Parameter*	*Name*
Powersource	Capacity	cap
Mechanical System	No. of Motors	nMot
Mechanical System	Wheel Dimension	rWheel
Mechanical System	Total Mass	mTot
Thermal System	Climatisation Power	hP
Thermal System	Trailer Cooling Power	cgP

Process and simulation-based Decision Making

Figure 5 summarizes the seamless integrated, collaborative workflow using MBSE principles and virtual testing to verify architecture performance. SysML models are created by formal processes following a strict R-F-L methodology to decompose the complex system of interest, (Aleksandraviciene and Morkevicius 2018; Weilkiens

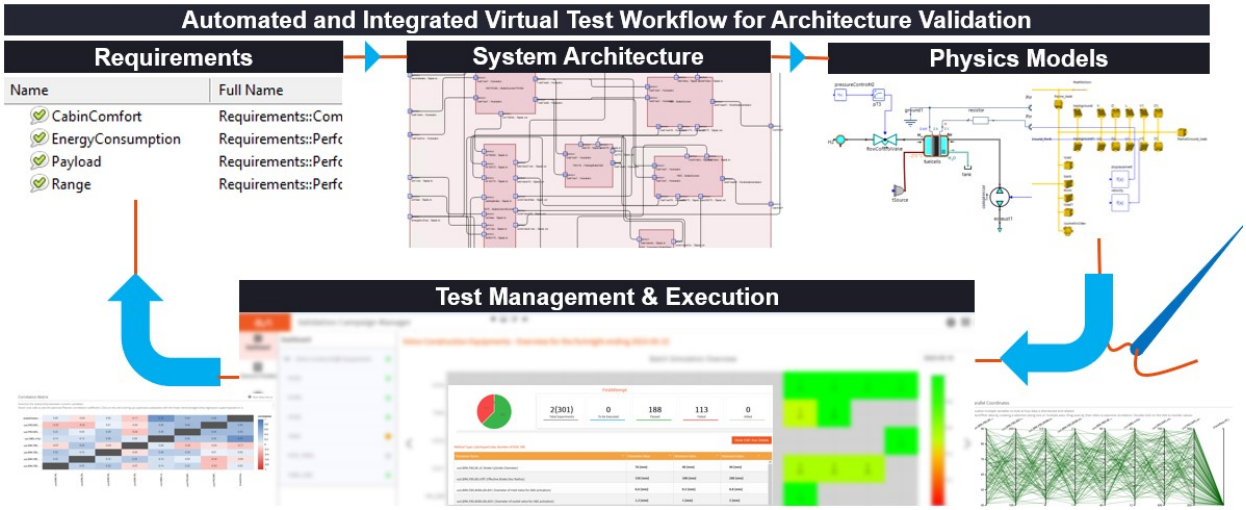

Figure 5. Integrated digital thread workflow spanning MBSE disciplines from requirements to architecture to physics for simulation and application of the physical models in virtual testing scenarios for early stage validation and design decision feedback

2014). In such hierarchical approach, lower level requirements and specifications are derived from higher level engineering results (e.g. *system* L defines F on *subsystem*, *subsystem* F defines R on *components*, etc). Applying "shift-left" paradigm by virtual design (with increasing model capabilities and fidelity along the process) enables decision making based on simulation results and the automatic propagation of top level stakeholder needs throughout the development task. In particular, use cases in this publication like

1. Architecture exploration for electrical system (battery or fuel cell) and thermal system (cabin heating with resistor or heatpump) with available or derived simplified models e.g. transfer functions, validating against high level performance indicators defined by the requirements above

2. Scenario-based design optimization regarding sizing and performance of subsystems on previously selected architecture level, e.g. number of drive motors in mechanical system, or deriving geometrical parameters and requirements for subsequent 3D design (CAD)

3. Operation and mission optimization for a given/ frozen design of the EV for in-service phases of the system

4. Prediction of performance degradation after 5 years or 2.000 cycles in operation, e.g. using aging models of power source system, and verification against lifecycle requirements

will benefit from continous artefact integration (data) and workflow automation, towards future AI[5] supported engineering business. Further downstream, to enable a traceable application of the generated physical simulation models with respect to scenario-based architecture and system performance validation, relevant information for this purpose is reused from the abstract systems model in SysML. More specifically, standard systems engineering entities like structural diagrams (L) shown in Figure 5, are augmented by stereotyping to specify the nature of physical connections between the different elements (*blocks*) and describe the architecture of the design system. Their allocation to certain functions (F) allows a filtering of components throughout the model generation in the simulation tool, hence supporting the focus of dedicated engineering teams. With the *verification* link given in requirement diagrams (R-L), specific parts of the system architecture can be applied in dedicated virtual testing models representing a specific scenario described by the linked requirement. As shown in Figure 6, this way, the physical design model (P), realizing the logical structure of the architecture, and the requirement (R) become ingredients of the actual testmodel. This testmodel generation is semi-automatic by instantiating the corresponding design component (as "Unit under Test" *uut*) that is linked to the requirement, and ensures traceability along the artefacts generation. Since requirements are often not formalized but given in natural language, the design model augmentation by stimulation and evaluation corresponding to the requirement is manual modeling effort. However, in previous studies, these aspects have been already automated by applying standardized interfaces for test automation[6].

As already mentioned above, the aspect of *configurations* of a product becomes more and more important in engineering workflows incorporating modeling and simulation shown in Figure 5, particularly for complex cyber-physical systems OEMs or suppliers have to handle full

[5] artificial intelligence

[6] ASAM XiL

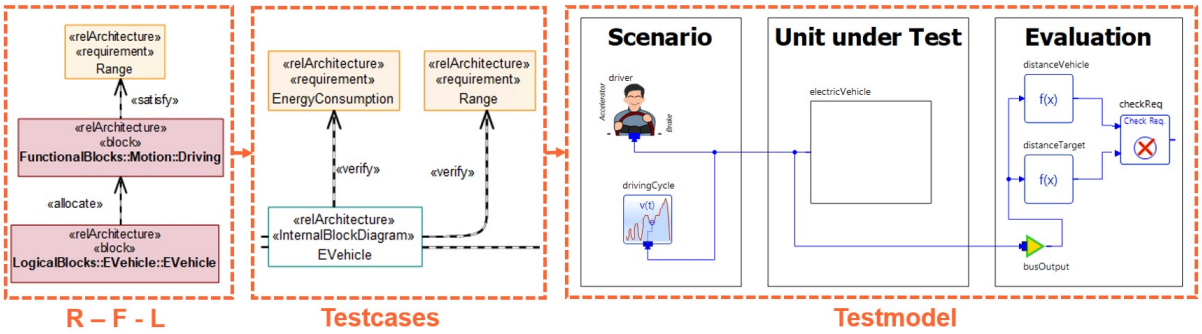

Figure 6. Concept to reuse tracelinks between SysML artefacts to generate testmodels in Modelica. Allocation to system functions allows filtering (left), verification links between logical layer and requirements (center) allow augmentation of design models with corresponding stimulation and test verdict (right, *failed* test) for automatic virtual testing processes

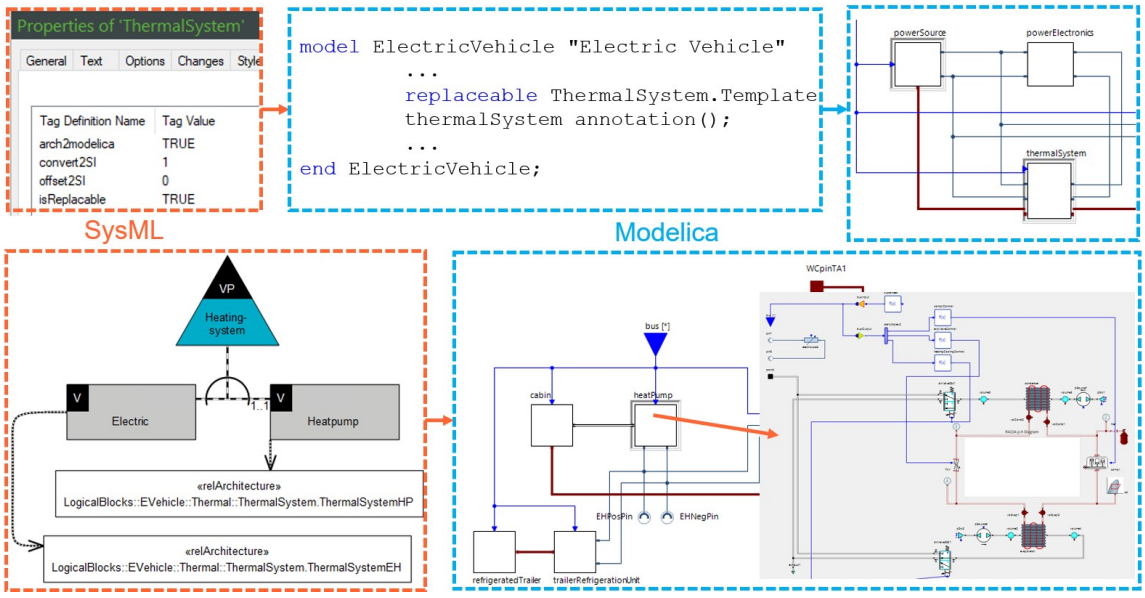

Figure 7. Automatic cross-domain transfer of product configuration specification by utilizing SysML entities like *stereotyping* (top left) and *variant diagrams, decision sets* (bottom left), and the Modelica *replacable* mechanism (right), to describe different architecture or component implementations, demonstrated on the *Thermalsystem* and the inner, selected *Heatpump* structure

product families. The need for cost reduction drives the modularization and identification of commonalities in design, and continous integration is a measure to achieve that. Both, architecture and simulation provide techniques to enable cross-domain transfer of such information, see Figure 7 for an overview on the implementation. Two main aspects need to be handled:

1. Express the variable component in the SysML model (L) by stereotyping and tagging as shown top left and transfer that information to the corresponding, automatically generated Modelica template (P) using the builtin replacable/redeclare mechanism in a certain instance of a model, e.g. for testing, as shown for the *ThermalSystem* block

2. Express the various, potential configurations of the system of interest as shown bottom left for the *Heatpump* or *Resistor Heating* implementation of the *ThermalSystem*, SysML tools offer dedicated artefacts like *variant diagrams* and *decision sets* to represent *variability* in a certain system, such expressions are standardized by ISO 26550 and variability or variants modeling is becoming a crucial part of SysML v2 description language (Object Management Group 2023), potentially leading to further enhanced capabilities and automation in this regard

In addition to the systems structure, the design parameters, that are defined on architectural or systems engineering level and attached to the different blocks/components (L), need to be handed over to simulation domain. Such parameters, like in Table 1, have to be used for testcampaign definition to reflect the design space definition for each physical implementation (P). Moreover, considering the corresponding parameter ranges in an automated fashion throughout the test description serves the use case of estimating the impact of real product deviations or toler-

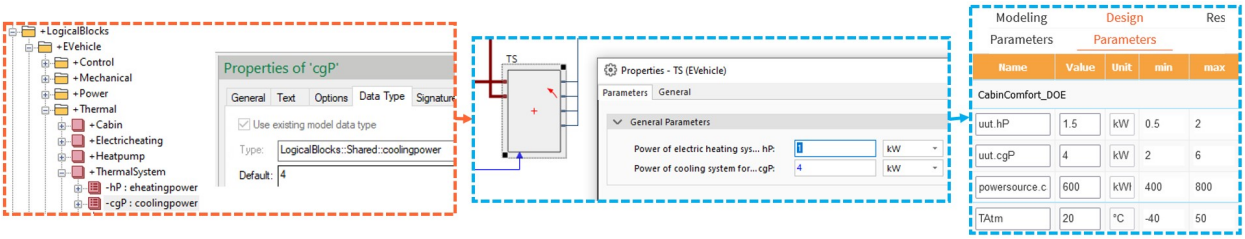

Figure 8. Automatic cross-domain transfer of global design parameters (default value, ranges and units) defined on architectural level by utilizing SysML entities like *Block Property* (left), and corresponding Modelica implementation (center) to reuse in VCM for experiment definition (right), visualised for the *Thermalsystem* verification

ances ("as manufactured"), to achieve a robust design right from the beginning. Apart from such conventional use of parameters for system or component sizing, the Modelica language offers another beneficial option for the application of architecture and structural decision making. Integer values on logical blocks are used to describe the number of instances of a same component inside that block, without changing the architecture with respect to physical domains of interfaces. A visual example is the *No. of Motors* parameter in Table 1, where the external structure and connectivity of the mechanical subsystem does not change, but the desired number of electrical machines connected to the outside interface (*multiplier*) is transfered from systems engineering level to simulation and vice versa. Optimization tasks like "Is one big motor better or worse regarding overall energy consumption compared to two smaller ones?" are significantly improved with better traceability and user convenience across the different development domains.

Again, the standard mechanisms of architectural and physical modeling are used to express and transfer the desig nparameters, their description, default values and ranges, Figure 8. In a continous integration implementation, the test management and execution system in Figure 5 lists these parameters automatically, to be used for the definition of the different experiments/ campaigns, either single runs or multiple variants simulations in applications like design space exploration or optimization (design of experiments).

Since the different architectural components discussed above, are linked to corresponding requirements that they should fulfil, Figure 6, the information of test cases which require a simulation model is automatically provided to the simulation engineer in the physical modeling tool, see Figure 9. Similar to the template generation of design models shown in Figure 3, the testmodels will be automatically generated by instantiating the correct modeling component when the user selects a test case in the list.

These features highlight the contribution of cross-domain integration to the scope of more efficient and reliable, collaborative engineering workflows. Aside of that, it should be emphasized again, that the underlying linked data ensures strict traceability, and continous integration allows

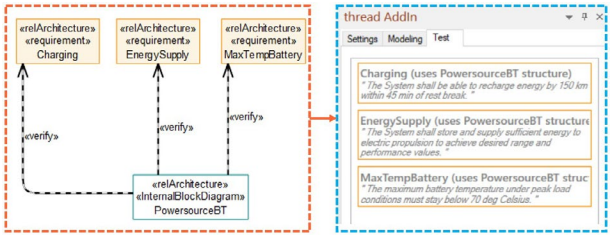

Figure 9. Representation of requirements that need to be tested by performance models in SimulationX Addin (right), defined by *verifies* links on architectural models in SysML (left), shown for the *Powersource* subsystem, see also Figure 6

for automatic top-down change transfer along the process. Changes in parameters or architectural structure are propagated and trigger model regenerations. On the other hand, fully bi-directional automatic bottom-up updates, like from 1D simulation results to architectural changes, are ususally not allowed in real development processes of complex systems, as they require impact assessment on higher, holistic system level.

Commercial Electric Vehicle Example

With respect to page limit, the technical engineering solutions described above, will be visualised by one specific example of a long-range truck design and sizing. Based on the explained high level requirements, system simulation is used to verify the performance of different, potential system architectures. This is a two-stage process, where both, the architecture evaluation and the following system and component sizing are verified on a detailed mission simulation. Such inherent incremental maturity rise along the development means, that the structure (L) of the system determines the functions (F) to be provided by the subsystem (and so on), hence lower level requirements are derived based on higher level simulation results. Close to state of the art parameter settings of heavy electric trucks with cooled trailers have been applied for the simulations below, as they became available and published recently.

Saying that, the scenario setup for the "architecture exploration" use case consists of two conditions for summer (30 °C) and winter (-15 °C). As mentioned above, it should be investigated which configuration of the

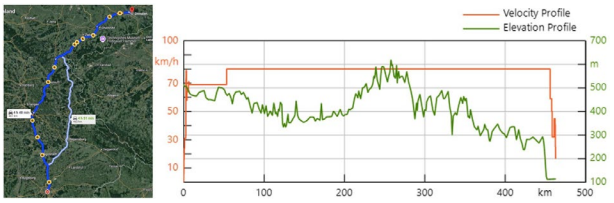

Figure 10. Examplary routing from Munich to Dresden with average velocity and elevation profiles exported by a route planner

- *PowerSource*: Battery vs. Fuelcell
- *ThermalSystem*: Heatpump vs. electrical Heating

would be the best choice with respect to the decision criteria of specific energy consumption for driving as well as climatisation of cabin and trailer. The actual models in the different domains have been created as explained and visualised above and considering the resulting test matrix, 8 simulations have been executed. Please note, that the physical implementations of the systems and subsystems of interest (*PowerSorce* and *ThermalSystem*) are simplified at this stage. Usually, at these very early phases of design more detailed models are simply not available, or an enormous number of simulation runs is required within multi-dimensional design space exploration of complex systems and the computational performance has to be maximised. Moreover, it should remarked again, that the physical models that are plugged into the architecture template by means of FMI originate from different sources, or are represented by native Modelica components.

However, all "full system" evaluations are done using the exemplary, envisioned route for the truck shown in Figure 10. This is crucial, as the different system implementations have different efficiencies and capabilities, e.g. regarding recuperation. Hence, the route profile of the specific mission has significant impact on the selection of a certain architecture configuration. Relevant information from the route like elevation (inclination) and average velocity are imported to a "drivingCycle" block, compare Figure 6, in the testmodels to apply proper conditions for stimulation. Please note, the maximum speed for heavy trucks is limited to 80 km/h on highways. It should be also remarked, that the tested route has a direction, means that the results will differ when going in reverse direction. Such considerations are subject to common trade-off studies.

A qualitative analysis is given in Figure 11 for first step decision making. With respect to the overall specific energy consumption shown on top, it becomes obvious that the battery configuration is the better choice, independent of the ambient conditions. This is because of several reasons. The major driver is the efficiency of the fuelcell to convert the hydrogen into electrical energy, in comparison to the battery that stores the required energy

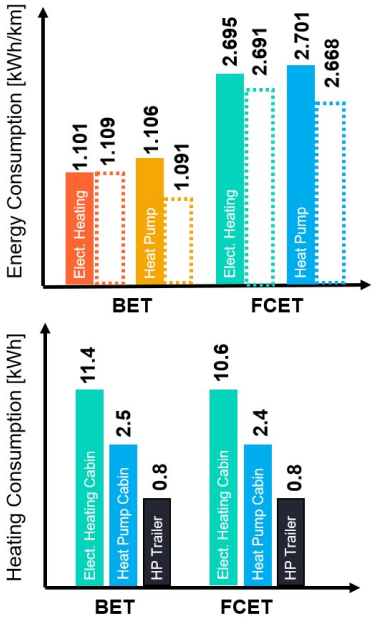

Figure 11. Qualitative comparison of performance numbers for the battery electric (BET) and fuel cell electric configuration (FCET), top: overall energy consumption for electric heating and heatpump configuration, each at summer (colored) and winter (dotted) ambient conditions, bottom: heating energy for the cold conditions, distributed over main consumers

directly. However, depending on the overall scenario, it might be necessary to extend the system boundaries and consider the costs and effort for external electrical energy (for charging) as well, which might in turn change the outcome as well. Aside of that, the energy comsumption in could conditions is lower for both configuration because of the lower demand by the trailer cooling system. Also, it should be remarked, that the electric heating cabin climatisation variant, seems to be slightly more efficient for the summer scenario, compared to the more complex heatpump system. However, this is simply to the fact that the eletric heating cannot cool the cabin in hot conditions, thus there is no energy consumption at all in this case, but it violates the functional requirement of "cooling" which is indicated by a *failed* test for *CabinComfort* requirement.

Looking at the *ThermalSystem* performance at the bottom of Figure 11, shows a clear benefit of the heatpump configuration in cold conditions, because of the much higher efficiency - as expected in this demonstration.

So, the overall architecture evaluation on this specific customer mission results in a decision for the battery-heatpump configuration. Based on this outcome, the detailed design of the specific subsystem implementations is taking place. The same architecture representation in the system simulation tool is used, but the simplified physical models of *Heatpump* and *Battery* are replaced by enhanced, more sophisticated model components, (Pukrushpan 2003; Hariharan, Tagade, and Ramachandran 2018), as shown in Figure 7. In this step, subsystem

Experiment Name	Experiment Status	Check Criteria	Required Value	Final Value
EnergyConsumption	Passed	checkReq.rf[-]	1	1
Range	Passed	checkReq.rf[-]	1	1
ClimbingPower	Failed	checkReq.rf[-]	1	0
Acceleration	Passed	checkReq.rf[-]	1	1

Experiment Name	Experiment Status	Check Criteria	Required Value	Final Value
MaxTempBattery	Passed	checkReq.rf[-]	1	1
CabinComfort	Failed	checkReq.rf[-]	1	0

Experiment Name	Experiment Status	Check Criteria	Required Value	Final Value
Recuperation	Passed	checkReq.rf[-]	1	1
Charging	Passed	checkReq.rf[-]	1	1

Figure 12. Test execution and requirements fulfilment overview in VCM triggering design decisions, models in the *driving performance* domain (top), models in the *thermal performance* domain (center), and models in the *electric performance* domain (bottom) for a given, exemplary parameter setting

and component requirements are derived incrementally throughout test execution with the detailed models, requirements on charging features that depend on the type of architecture.

Figure 12 provides an overview of test results (with respect to *passed/failed* verdict) using the configuration identified above, with a certain, exemplary setting of design parameters selected from the given ranges as shown in Figure 6 (right). It can be seen, that most of the requirements are met by the design in the described mission. However, the *ClimbingPower* and *CabinComfort* tests are failed. The analyis requires a more detailed view into the transient behaviour of a certain simulation, to identify the root cause. As an example, the latter testcase is executed in Figure 13 in a "design of experiments" run, to figure out the impact of input conditions and design parameter values. It can be seen, that some, mainly the coldest ambient conditions, do not fulfil the requirement of a cabin climatisation between 18 and 25 deg Celsius (left) for the particular parameter setting. The VCM provides various analysis and data analytics tools and capabilities. With the parallel coordinates on the right, the ranges of relevant parameters (here the *climatisation power hP* and the *trailer cooling power cgP*) can be limited to valid combinations, that satisfy the *CabinComfort* requirement. Such evaluation will be feed back into the systems engineering design phase and can be further automated, for applications like functional optimization.

Once the ideal component design and parameter setting regarding the different requirements are identified, the configuration can be frozen and used for detailed transient analysis and load case generation for design steps in the 3D geometry domain downstream the V-cycle, see section below. In example, Figure 14 visualises the evolution of battery charge, recuperation power and mechanical braking action along the route (excerpt), testing the *Range* and *Recuperation* requirement. Aside of that, further engineering tasks like mission optimization, or operational predictions like aging of battery power source are supported by the digital twin character of the architecture simulation models.

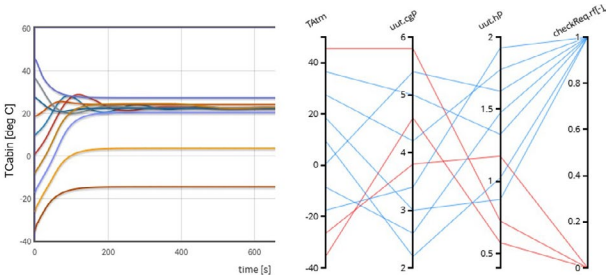

Figure 13. Example of visualisation in VCM from the *CabinComfort* testcase executed at 8 different ambient conditions for demonstration of engineering tasks like design optimization and parameter identification, transient results left, parallel coordinates to narrow down valid parameter ranges within design space satisfying the requirement (blue)

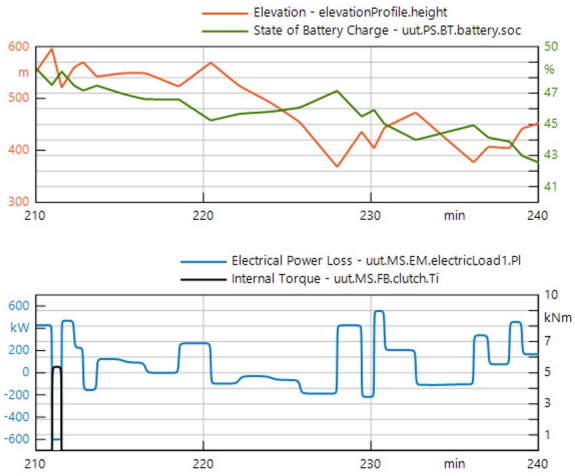

Figure 14. Example of a zoom into transient results from the *Range* testcase running the mission in Figure 10 for design optimization, bottom diagram indicates mechanical breaking action when maximum recuperation is achieved on steep descents

Finally, it should be mentioned that the results and derived architecture and components look different for the system of a city bus because of the different scenario. However, the major benefit of such continous integrated design process, is the agility to quickly identify new architectures on changed requirements.

Reuse of Architecture Performance Models

As mentioned above, the simulation results gained in the demonstrated process will serve as inputs further downstream the product development cycle, for geometrical definitions (system sizing) or providing load cases for

Figure 15. Example of 1D physical model (ABS brake) integrated into 3D FEM model by means of FMI for realistic pre-crash simulation

detailed 3D CAE analysis tasks. Aside of these natural cross-domain interactions, particularely the Modelica models generated for performance validation of system architectures or subsystems can be reused as FMI integrated components, for sophisticated, more realistic, scenario-based 3D FEM simulations, as shown in Figure 15. With such enhanced coupling, certification credits for crash simulations on safety critical battery and fuel cell architectures (e.g. ISO 23273) in the challenging EV domain are enabled and already demonstrated.

4 Conclusion

Establishing continuous and agile workflows in design and validation of complex cyberphysical systems, by enabling collaboration on heterogeneous tool and stakeholder ecosystems in early phases of development, leverages the potential of ongoing digitalisation as shown in the present demonstration. It addresses currently existing process and traceability gaps between the engineering disciplines of requirements management, architectural design, physical development and virtual performance and puts system simulation in a broader, holistic system context. This way, 1D simulation evolves from an isolated activity, acting in a silo with well known issues and friction when it comes to integration, towards an integral part of virtual development applying and following model-based systems engineering methodologies to master present and future process and product complexity. With its ability of serving as model aggregator, Modelica plays a crucial role in collaborative multipartner processes, examplary for complex systems, early stage validation.

With the presented digital thread implementation, not only design and verification becomes more efficient, reliable and collaborative, but also sales engineering tasks like RFP phases (request for proposal) benefits from much reduced task cycle times. The cross-domain variability support allows the fast and reliable selection of the best configuration in a complex product family for a customer specific mission.

In a next step, the architecture identification can be automated by enabling experiment definitions on component implementations. With the reuse of dedicated SysML artefacts (variant diagram, decision set, etc) in the test management system (e.g. VCM) design space explorations can be easily executed and evaluated. This would enable further AI support by simulation-based decision making towards more autonomous processes.

Finally, with the upcoming layered standard on *SSP traceability*, further automated test model generation with respect to stimulation and evaluation of performance models, based on reusable meta data for test specification, is expected.

References

Aleksandraviciene, Aiste and Aurelijus Morkevicius (2018). *NoMagic 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.

Douglass, Bruce Powel (2016). *Agile Systems Engineering*. 1st ed. Morgan Kaufmann Publishers. ISBN: 978-0-12-802120-0.

Gottschall, Marcel, Bastian Binder, and Alexis Castel (2022). "Towards Certification by Simulation with model-based continuous Engineering Processes showcased on eVTOL Application". In: *78th VFS Forum and Technolgy Display*. Vertical Flight Society.

Hariharan, Krishnan, Piyush Tagade, and Sanoop Ramachandran (2018). *Mathematical Modeling of Lithium Batteries*. 1st ed. Springer. ISBN: 978-3-319-03526-0.

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 (2022a). *Functional Mock-up Interface for Model Exchange and Co-Simulation Version 3*. Tech. rep. Linköping: Modelica Association. URL: https://fmi-standard.org.

Modelica Association (2022b). *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/.

Object Management Group (2023). *OMG Systems Modeling Language Version 2 - Language Specification*. 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/.

Pukrushpan, Jay Tawee (2003). "Modeling and Control of Fuel Cell Systems and Fuel Processors". Doctoral dissertation. The University of Michigan, Department of Mechanical Engineering.

The INTO-CPS Association (2018). *The INtegrated TOolchain for Cyber-Physical Systems: a Guide*. Tech. rep. INTO-CPS. URL: https://into-cps.org/fileadmin/into-cps.org/Filer/INTO-CPS-Manifesto.pdf.

Weilkiens, Tim (2014). *Systems Engineering mit SysML/UML*. 3rd ed. dpunkt. ISBN: 978-3-86490-091-4.