Objectively Defined Intended Uses, a Prerequisite to Efficient MBSE

Erik Rosenlund[1,2] Robert Hällqvist[1,2] Robert Braun[2] Petter Krus[2]

[1]Saab Aeronautics, Linköping, Sweden
[2]Fluid and Mechatronic Systems, Linköping University, Sweden

Abstract

This article proposes a method for improved model verification within Large-Scale Simulators (LSS). The approach relies on machine-interoperable traceability of model verification information, such as model Operational Domains (ODs). This enables automated evaluation of model relevance and facilitates the combination of models for a broader evaluation of credible simulation results. The paper introduces a proof-of-concept testbed for verification of black-box models against model requirements. Furthermore, the results also include a proposal for a machine-readable format to capture model requirement Verification & Validation (V&V) results, along with the resulting model and updated model OD information.
Keywords: Verification and Validation, Operational Domain, Large-Scale Simulators (LSS), Machine-Interoperable Traceability, Model Reuse, Model Exploration, Simulation Credibility, FMI, SSP, SSP Traceability

1 Introduction

In the area of modeling and simulation, the primary challenges no longer concern whether something can be simulated or not but rather if the results are credible and can be utilized as intended. Many different factors influence whether the results can be used for a specific purpose or not, several of them exemplified by *NASA STD-7009* (2008):

- Are models validated for the scenario simulated?
- Does the model fidelity correspond to what is required by the intended use of the results?
- Are aggregation effects between interacting models sufficiently accounted for?

For a user to make credible decisions based on simulation results, all questions above, and many more, must be answered to the rigor required purpose. When scaling from a scenario where the model designer is making decisions based on simulation results, to a use-case where the simulator end-user has minimal knowledge of the simulated models, additional information needs to be transferred along with the model for the user to be able to draw conclusions regarding the credibility of the test results. Increased simulator complexity or model complexity increases the need for this knowledge transfer. Simulator complexity is in this context seen to be dictated by factors that increase the number of support functions needed for a simulation to execute; such as functions that enable the mixing of Software-in-the-loop (SIL) and Hardware-in-the-loop (HIL) or distributing computations over multiple computers.

How this additional information is to be transferred is situation-dependent, for smaller projects or Modeling & Simulation (M&S) activities it could be enough to discuss model requirements between the user and model designer. For *LSS* (Andersson 2012; Steinkellner 2011), the number of models and support systems makes the amount of knowledge needed to evaluate if results are credible almost impossible for a single user to manually ingest in a reasonable time frame. Thus solutions to automate this workflow are required (Hällqvist 2023). This creates additional demands on the model designer and many model requirements that previously were implicit now need to be explicit and verified in a traceable way. Examples of model characteristics that typically fall into this category are related to runtime performance, numerical errors, or end-use not captured in the original model specification. The gain of explicitly expressing these requirements, and imposing a standard for how the information is relayed, is seen as an enabler in further scaling of LSS.

In short, LSS imposes an increased need for standardized knowledge transfer and evaluation of model intended uses. Enabling this effectively is stipulated to minimize the need for users to possess detailed knowledge of all constituent models and their respective implementations. This would allow extended simulator utilization and increase the credibility of decisions based on simulator results. The goal of this work is to propose a machine-interoperable way of providing model requirement verification information. This goal is expressed in the following research questions,

> RQ: How can information regarding model verification activities be communicated in a tool-independent way to enable model evaluation in a LSS environment?

1.1 Contributions

The work includes the development, evaluation, and demonstration of a proof of concept testbed that provides a structured way of verifying black-box models against requirements. This testbed enables the verification of legacy models where knowledge of the model's intended use and model requirements are documented according to the traditional document-centric paradigm; any improvements on this topic are highly desirable in organizations with a large knowledge capital expressed in legacy models. The testbed verification results are documented in a machine-readable format, based on Extensible Markup Language (XML). The proposed format provides a container for mediating information regarding model verification results that enable automated reasoning on how models can be combined without implicit compromises of credibility. Integrating this information into the model enhances availability and traceability, while also demonstrating an industry-relevant application of the SSP Traceability (Modelica Association 2022) format Simulation Resource Meta Data (SRMD). The proposed structure is thought to initiate a discussion on the establishment of an end-user standardized layer on top of SRMD, capturing what meta-data is relevant to encapsulate together with a model or set of coupled models.

1.2 Research Method

The utilized research method is built on the established method, "Industry-as-laboratory" proposed by Potts (1993), where the industry provides the questions and then acts as a base for conducting experiments. The goal is to enable the study of real-world problems in a scientific manner. The method has since been further refined by Muller (2020) and has previously been applied successfully within the field of aeronautical engineering, see for example Eek, Hällqvist, et al. (2016) and Oprea (2022).

2 Theoretical Background

2.1 Verification and Validation

There have been many attempts in both academia and industry to clarify the difference between model verification and model validation (Wang et al. 2019); however, should we strive to separate the two different sets of activities? Model verification is popularly defined as the quest to answer the question of whether the model is built "right" whereas validation is the quest to answer the question of whether the right model has been built (Osman Balci 1997). In other words, verification concerns a "quality control" on conducted model transformations typically associated with the identification of "bugs" in the implementation. In contrast, validation concerns ensuring that the model is fit for its purpose. So, model validity needs to be assessed with respect to the purpose of the model. Verification does not. However, transforming model requirements into a model is a transformation that needs to be verified, just as the transformation from source code to an executable model that can be integrated in a simulator. In that sense, the modeling begins with the requirements. Any model purpose is therefore ideally expressed explicitly in the form of requirements to, among other things, simplify the model development (Hällqvist 2023). Examples of such functional requirements are shown in Enumeration 1.

Enumeration 1. Functional requirements of a model.

1. The model predicting the ambient air temperature shall predict the temperature with 95% accuracy, concerning the corresponding physical system, in its complete input space, see Section 2.3.
2. The model predicting the ambient air shall relate the model inputs altitude and speed to its response quantity in an *International Standard Atmosphere (ISA)* (*ISO 2533* 1975) atmosphere.

These two requirements can act as a solid foundation to a *model development process* (Carlsson et al. 2012). Transforming model requirements into a mathematical model should be accompanied by verification activities. Such verification activities can be done without any subjective judgment; however, they cannot be concluded until a 95% accuracy in the model response quantity, according to requirement one in Enumeration 1, can be guaranteed with sufficient probability to deem the risk of concluding the activities as acceptable. In early phases, this can be achieved through, for example, Uncertainty Quantification (UQ) (Riedmaier et al. 2020) techniques.

In later phases the accuracy can be accessed through comparisons against in-situ measurements (Beisbart and Saam 2019; Sargent 2010), accompanied by UQ analysis if deemed necessary. Once this point has been reached, there is no need to initiate any activities denoted as "validation"; As the model has been deemed to fulfill its specified purpose (the two requirements). With this line of reasoning, model validation is a subset of verification concerning only the verification of requirements that are implicit or require subjective judgment. These types of requirements are, as pointed out earlier, undesirable. Consequently, we should strive towards declaring an intended use free from implicit requirements and remove the need for model validation.

2.2 Intended Uses

Expressing explicit modeling requirements with verification criteria is a challenging task that requires substantial research. Murray-Smith (2019) emphasizes how important it is for a user to be aware of model limitations and accuracy for all of its intended use and that formal testing is often lacking during model development. Methods and tools to aid engineers in this process are essential. Hällqvist (2023) partitioning model purposes into two different categories: coarse-grained intended-uses and

fine-grained intended-uses:

A coarse-grained intended use qualitatively expresses the purpose of a M&S application in some format, formal or natural language, with a clear connection to, if needed for acceptance, one or more fine-grained intended uses.

A fine-grained intended use quantitatively expresses the purpose of a M&S application in a formal format, with a mathematical connection to one or more validation or predictive capability metrics, predictive capability defining a notion of capturing model representativeness.

To concretize requirements as fine-grained intended uses and to utilize these as foundation during verification is to provide increased traceability during the model design process. Correlations can be made towards Test-driven development (TDD) and maybe even more so towards Acceptance test–driven development (ATDD) where acceptance tests are written before feature development. As described by Martin and Melnik, this allows the designer to analyze the requirements, in a structured way, and to evaluate how they can be translated into tests (Martin and Melnik 2008). Verifying this translation should be done by both the *architect* and *model designer* to ensure that requirements are interpreted correctly and test cases cover all intended uses (Pugh 2010). Expressing a requirement as a test can be one of the most effective ways to verify its 'completeness and accuracy' and the process can be utilized to weed out implicit or ambiguous requirements thus reducing the risk of not 'designing the model right'.

2.3 Operational Domains

The work on concretizing model ODs is viewed as an essential part of expressing fine-grained intended uses. A model OD is viewed as an enclosed n-dimensional volume representing the model's feasible input space. A sought, or required, model OD could be seen as an outcome of the model specification activity. The modeler then has a challenge in realizing or identifying existing models capturing, selected aspects of the physical system to be modeled. Three such ODs are schematically visualized in Figure 1. The OD denoted $OD_{Model\ B}$ schematically represents the feasible input space of an existing legacy model of the System of Interest (SoI). The, by the M&S task, required model OD is visualized as $OD_{Model\ A}$. A first verification activity could encompass the evaluation and comparison of $OD_{Model\ A}$ and $OD_{Model\ B}$; where $OD_{Model\ B}$ can have been deduced analytically or empirically. This verification will concern iterative negotiations, between the model end-users and developer, regarding the overlapping regions of $OD_{Model\ A}$ and $OD_{Model\ B}$ to deduce if the developed model is fit for purpose.

A frequently used representation of the n-dimensional volume representing the OD has been that of an n-dimensional hypercube constrained by the minimum and maximum value of each input variable (FMI development group 2022). A hypercube representation may in certain cases result in a loss of information regarding how a model can be used, see Figure 2 where any attempt to limit the OD to a rectangular domain will reduce the representation of the OD with respect to the actual capability of the model. Both Roy and Oberkampf (2011) and Hällqvist, Eek, et al. (2023) utilize n-dimensional convex hulls to represent ODs; however, they are still models of the domain and suffer from the same problems that any model representation incurs, mainly, 'is it good enough?' Nonetheless, this approach enables transferring a more nuanced picture of the actual domain at the cost of simplicity." The documented use, in the context of the presented research, has focused on creating and utilizing hulls for verification, but not on how it can be stored for further reuse downstream in the M&S chain.

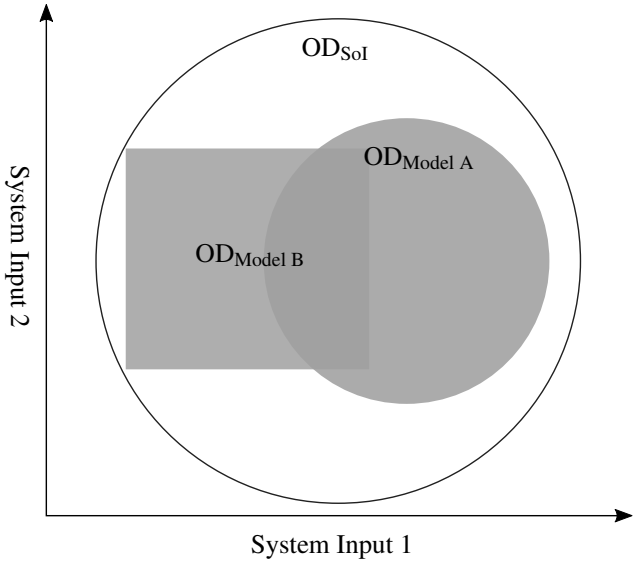


Figure 1. Schematic description of two-dimensional ODs: OD_{SoI} represents the system of interest input space, and $OD_{Model\ i}$ the OD of two different models representing different parts of the system of interest input space.

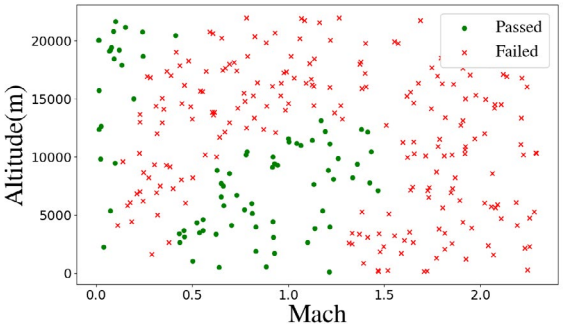


Figure 2. Verification samples mapping input space towards passed or failed evaluations.

2.3.1 Delimitations

Multiple solutions for passing information regarding verification results exist, the most straightforward one is to provide the coordinates of all the tests. This leaves the full responsibility of interpreting the verification results to the end-user. In certain situations, this may be preferred but in LSS this is not an option due to the broad user base and extensive quantities of packaged information. In the end, the choice should be the least complex solution that encapsulates sufficient information to express the model requirement. For example, whilst potentially providing a higher fidelity representation of a model OD, a concave hull is more complex to construct and utilize than a convex hull. Where a set of points can only have one solution in a convex volume, the number of solutions in a concave hull grows rapidly with the number of points (Asaeedi, Didehvar, and Mohades 2014). Other solutions such as clustering algorithms to map the OD into smaller regions of hypercubes, hyperspheres or hulls may provide a more detailed representation, but the choice of clustering algorithm must then be taken into account leading to increased complexity. We will acknowledge the existence of other solutions and limit the scope to the shape of the transferable n-dimensional volume as a convex hull or one of its simpler representations such as the hypercube, since the shape itself is not vital in establishing the methodology around its transfer and it has been utilized in a similar context in related research.

2.4 Model reuse and Traceability

Many of the aspects of model reuse can be correlated to the Findable, Accessible, Interoperable, Reusable (FAIR) principles (Wilkinson et al. 2016) and their role in enabling increased reuse of datasets or, to some extent, Long term archiving and retrieval (LOTAR) (Coïc et al. 2021) in its goal to provide a common standard for geometry data. All aspects of FAIR are needed as enablers for credible model reuse. The Functional Mock-up Interface (FMI) (FMI development group 2019) and System Structure and Parameterization (SSP) standards (Modelica Association Project System Structure and Parameterization 2019) enable some of the FAIR principles by providing common interfaces and data structures, enabling model reuse over a multitude of tools and simulation purposes. However, they do not capture all aspects concerning the *Findable* principle of FAIR. While there are mechanisms to convey information concerning units and permissible input ranges for models or systems, more comprehensive information of intended use is not supported in a structured way as of now (FMI development group 2022).

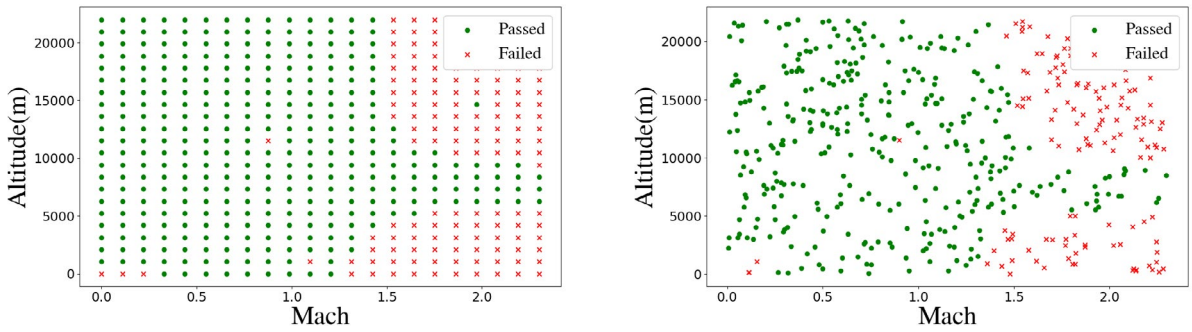
A model expressed in the form $\vec{y}_t = F(\vec{x}_t, \vec{x}_{t-1}, ..., \vec{x}_0)$, often referred to as a *computational model* (P. Fritzson 2004; Ljung and Glad 2004), adds something that static data does not; a data conversion or a *predictive capability* (Beisbart and Saam 2019). This creates an additional requirement that FAIR does not encompass, traceability when it comes to results. Within the healthcare sector, Erdemir et al. (2020) states that having the ability to associate results to input data and model version is "critical for accurate interpretation, repeatability, reproducibility, and debugging of the simulation predictions". There is no reason that this should not be true for the field of complex product development.

The area of storing and reusing engineering knowledge has been under intensive research for a long time (Robinson et al. 2004; Sivard 2001) and according to Pokojski, Knap, and Skotnicki (2021) any knowledge from subject matter experts that can be stored and reused will be beneficiary. Traceability between the executable model and the founding model requirements is one of the cornerstones when it comes to model use; any use of a model in a context where it can not be proved to be credible is by definition not credible (O. Balci and Ormsby 2000). A multitude of standards dictate how requirements are to be traced through a product life-cycle, e.g. *ISO 26262* (2018), *DO-178C* (2012), *NASA STD-7009* (2008) or *MIL-STD-3022* (2008). Since models and the resulting data often are not part of the end-product they can sometimes be exempted from mentioned standards. However, lacking such information traceability can impede the possibility of model reuse. It additionally restricts the utilization of models employed in product verification (Oprea 2022; Hällqvist 2023) or where models are included in the end-product, such as pilot training simulators (*Gripen Mission Trainers* 2024). Transparency and traceability of any data underlying decisions and a formal testing process increase the credibility of models greatly (Murray-Smith 2019).

Increasing utilization of simulation results in the product development process prompts harsher requirements regarding the traceability of models and simulation results (Level 2024). This is the reason for introducing the "Credible Simulation Process Framework" within the Simulation-based Engineering and Testing of Automated Driving (SET Level) project (Level 2024). The process is a base for the SSP Traceability standard (Modelica Association 2022).

2.4.1 Delimitations

To convey model ODs, the SRMD container of the SSP Traceability is selected as an initial bearer. Its intent defined as “SRMD files are used to define essential metadata for resources that can help users quickly understand the content and intent of a simulation” (Modelica Association Project System Structure and Parameterization 2019) goes well in line with the intent to store information regarding intended use. The format sets few boundaries to what can be stored within the container and it provides large freedom in how the resulting OD can be expressed. When looking at information abstraction levels, SSP is more at-



(a) Grid search, evenly distributed points over the input OD. **(b)** *Random search*, uniform distribution over the input OD.

Figure 3. Different methods used for model Operational Domain (OD) exploration.

tuned to system-level information, and FMI is closer to the model level. In some cases, it would be more fitting to utilize a layered FMI standard to convey the OD. This does however pose a limitation in usage, that the information can only be conveyed on a model level and not on a system level. To remedy this, and get access to both model and system standards, a simple solution of incorporating standalone models in a SSP is proposed. A single model can be viewed as a system and can easily be incorporated within a SSP container to get access to both FMI and SSP layered standards. It is possible to do the opposite but not without a loss of flexibility regarding the inner workings of the SSP and then only layered standards of FMI would be available.

2.5 Exploration

The primary method of model exploration involves exposing the model to various scenarios and evaluating the outcomes. Model exploration allows the *model designer* to verify the model against requirements, whether for new model design or model reuse. Many models are developed for, and subjected to, use where the quantities of interests are time-dependent, this is in contrast to scenarios performed under steady-state operating conditions. When synthesizing simulation scenarios the former will hence be referred to as dynamic scenarios and the latter as steady-state scenarios. The underlying purpose of model exploration utilized for verification differs somewhat from Design Space Exploration (DSE) or Hyperparameter Optimization (HPO); while the latter aims to find global extrema in the form of an optimal design, the former seeks to continuously increase the understanding of specific model behavior, often measured by a *coverage* metric (Atamturktur, Hemez, et al. 2009).

To empirically verify the model OD, various established methods in DSE and HPO are considered viable alternatives for model exploration. These methods include ***random search***, where values for each input variable are randomly sampled from a uniform distribution (see Figure 3b). In contrast, ***grid search*** systematically maps the entire input domain with evenly distributed points in a grid (see Figure 3a). Lastly, ***Bayesian search*** iteratively evaluates previous simulations to identify and further explore areas of interest (see Figure 4). This is often done by comparing the resulting *coverage* for new potential points.

According to Feurer and Hutter (2019), *black-box* methods such as *random search* and *grid search* suffer from the *curse of dimensionality* and can take a long time to search a multidimensional volume compared to guided search methods like *Bayesian search*. However, both *random search* and *grid search* are simple and straightforward in their implementation, requiring minimal tuning compared to *Bayesian* algorithms. Comparing *random search* to *grid search*, in x explorations, *random search* will evaluate x different values, whereas *grid search* will only explore $x^{1/n}$ different values since each row corresponds to the same exploration value (Feurer and Hutter 2019). For input combinations with low correlation, this can result in simulations that do not contribute new information. *Random search*, unlike *grid search*, is also an *embarrassingly parallel* algorithm (Herlihy and Shavit 2012), making it highly parallelizable with minimal overhead.

2.5.1 Delimitations

For this study, only steady-state scenarios will be used. These scenarios are considered complex enough to yield useful results while allowing for straightforward evaluation. However, future research may benefit from expressing *coarse-grained* intended use as use-cases, as proposed by Andersson and Carlsson (2012), to deduce dynamic scenarios and create variations used for verification.

Two different exploration methods are selected to verify the model OD: *Random search* and *Grid search*, primarily due to their ease of use. The assessment of coverage based on different metrics, as summarized by Atamturktur, Egeberg, et al. (2015), is also omitted from this study.

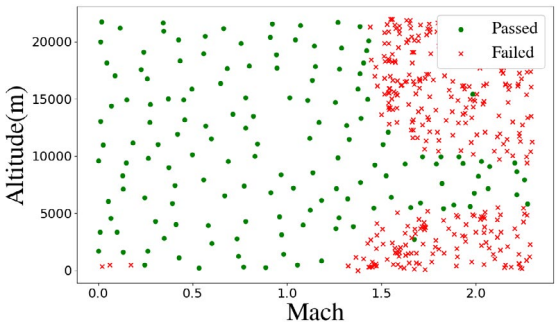


Figure 4. *Bayesian search*, placement of test coordinate depending on previous evaluations.

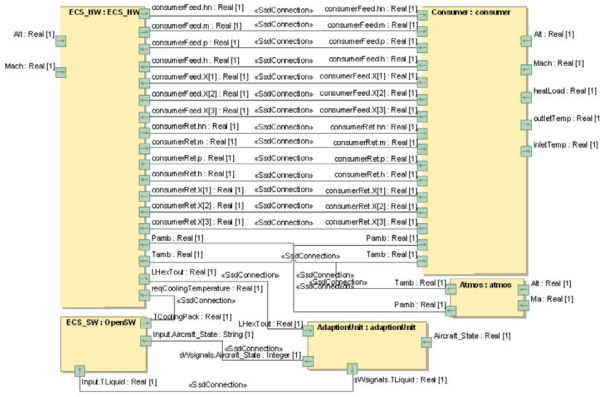


Figure 5. Application example architecture expressed as a SysML block diagram.

3 Application example

The application example simulator used in this study is a virtual reference system, originally developed in the EMBrACE project (ITEA 3 2019). This system is based on publicly available data regarding the modeled constituent sub-systems (Schminder et al. 2018; Hällqvist, Schminder, et al. 2018), and it offers a level of complexity and requirements similar to those encountered during the preliminary and detailed design of aircraft subsystems at Saab Aeronautics. The application example consists of a simple heat sink to consumer loop, including an Environmental Control System (ECS) and its controlling software, as illustrated in Figure 5. This system aligns well with the scope of the study and it provides opportunities to evaluate the use-case specified in Section 4. An initial OD volume is derived from the sub-system requirements. The OD is a subset of the full input space to increase results visibility; here limited to two input variables, altitude and mach. Other inputs are fixed to a specified value in collaboration with the *model designer*.

Two of the models in the application example architecture are chosen for this study: the ECS model and the model representing a generic consumer of cooling power. The inner workings of these two different models have been described in detail by Hällqvist, Munjulury, et al. (VI). In short, however, the ECS model includes both a traditional, bleed-driven, cooling system as well as a coolant distribution system. The coolant distribution system exploits a liquid coolant to transport heat from the consumer to a heat sink utilizing a modeled pump, a heat exchanger, and piping components all modeled using Modelica Standard Library (MSL), initially presented by Pop and P. A. Fritzson (2004), and the Saab in-house Modelica Fluid Light (MFL) (Eek, Gavel, and Ölvander 2017) Modelica library. The consumer model coupled to the ECS model is also modeled using components from the same Modelica libraries. However, the aircraft sub-systems that the models represent are typically developed by different organizations at Saab which motivates the co-simulation approach compared to developing a Modelica monolith.

4 Use-Case

The use-case presented in this section aims to exemplify the need for the research in focus. The use-case aims to capture a realistic and likely scenario in any organization applying M&S for decision-making, in any life-cycle phase. The presented use-case is seen to be applicable also in early life-cycle phases such as *concept development* (Raymer 2018; International Council on Systems Engineering 2015); however, the activities are likely less formal and rigorous to reflect the pace and information uncertainty inherent to the early phases.

4.1 Prerequisites

A need for a new model of a system, sub-system, or component emerges as a result of an engineering task deduced as most efficiently tackled through M&S. Additionally, a set of model requirements have been deduced, see for example Section 2.2, from the requirements on the physical system along with the M&S need. Several similar models exist, as a result of previous M&S activities, but their fit to the current model requirements is uncertain.

4.2 Actors

A total of three actors participating in the use-case are identified: the ***architect***, the ***model designer***, and the ***simulation engineer***. The *architect* supplies requirements on the physical system in focus and utilizes the M&S results to evaluate the system design. The *model designer*, in close collaboration with the *architect*, transforms system requirements to requirements on the corresponding virtual system implementation; furthermore, they design and verify the model against the supplied requirements. The *simulation engineer* utilizes the model to produce the results needed by the *architect*.

It is important to recognize that permutations of the above-stated roles may occur. Naturally, the setup decided to be most efficient in advancing the task at hand is the one that should be employed. The immediate need for trace-

ability of all information concerning the utilization of the model increases when the number of involved actors increases. However, please note that traceability to modeling requirements is here viewed as essential regardless of whether adopting a life-cycle perspective or not.

4.3 Main Scenario

1. The *architect* is to decide on a system design, and appropriate knowledge regarding a constituent subsystem is lacking.

2. The *architect* and the *model designer* identify a potential to fill this gap in knowledge by means of simulating the subsystem.

3. The *model designer* searches central model storage for possible models to reuse.

4. The *model designer* finds a model that could fit the requirements, but the documentation concerning the usage in the new context is incomplete.

5. The *model designer* explores the model to verify it against the new set of requirements. In each exploratory simulation, the model is evaluated using the steps presented in Enumeration 2 until a sufficient *coverage* (Atamturktur, Egeberg, et al. 2015) of the input space is achieved.

6. The *model designer* evaluates the results from 5. If the actor is unsuccessful, he or she adjusts the model and reiterates the main scenario steps from 5.

7. The *simulation engineer* utilizes the model when simulating the subsystem.

8. The *architect* utilizes the results as the basis for selecting, or refining, a system design.

Enumeration 2. Detailed description of the model evaluation steps.

1. Pre-processing, the test input point is selected from the initial OD using an exploration algorithm, and a steady-state scenario is synthesized from the values. (Figure 9)
2. Run simulation, the simulation encompasses three different phases, the duration of the phases is ideally objectively defined with a connection to the model characteristics. (Figure 6)

    A. Ramp-up phase, enables a smooth and numerically sound transition from a cold model state to the test point.

    B. Stabilization phase, allows the model to reach a steady-state at the test input point.

    C. Measurement phase, the model Quantity of Interests (QoIs) are recorded.

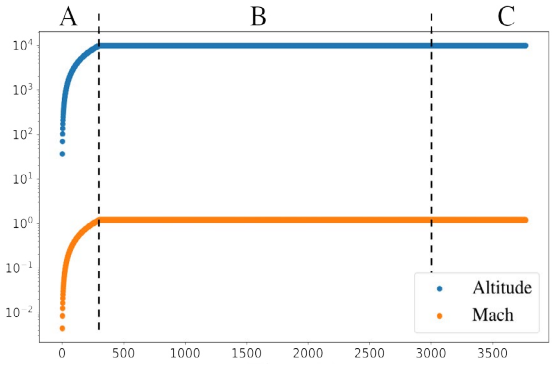


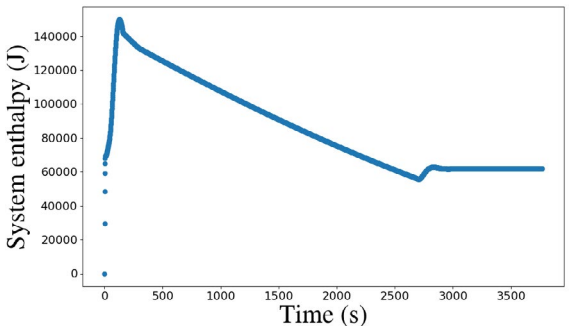
Figure 6. Synthesized simulation scenario of a set of input coordinates, i.e. input steady-state values representing one of the sets of coordinates of Fig. 9. The scenario is partitioned into three different phases: *A-Transitioning from cold to test coordinate*, *B-fixing the input to the selected test coordinate*, and *C-Evaluating test coordinate based on output measurements*.

3. Post-processing, the simulation results (Figure 7a and 7b) are evaluated with respect to the model requirements.

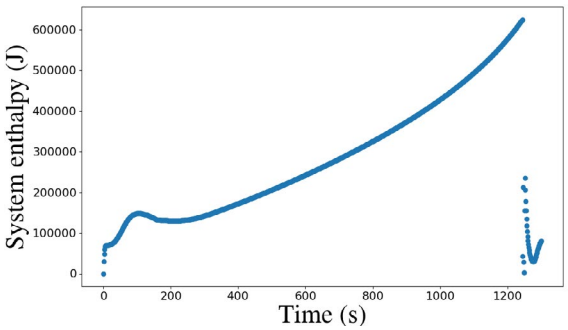
From the results of the model evaluation steps described in Enumeration 2, initial ODs in the form of hulls are generated. These reference hulls enable comparing OD between models and identifying limitations concerning the system's intended use. They also aid in identifying potential attributes for a standardized data format. The determination of sufficient coverage is left to the discretion of the model designer on a case-by-case basis.
The test framework DevelOpment, RIgorous, and automated assessment of models and Simulators (DORIS) (Hällqvist 2023), implemented in the now finalized project *Digital Twin for Automated Flight Test Evaluation and Model Validation* (Hällqvist 2019), is extended and utilized to evaluate models. The software is built around OMSimulator (Ochel et al. 2019) enabling simulations unified by Modelica Association (MA) (The Modelica Association 2019) standards such as FMI and SSP. It enables exploratory testing of models in an automated and repeatable way. The evaluation against requirements will be limited to check model robustness. A set of general model integration requirements is exemplified in Enumeration 3.

Enumeration 3. General model integration requirements adopted to exemplify the presented research.

1. The simulation shall not crash.
2. There shall not be any errors or warnings due to computational/solver problems.
3. Any single time step should be executed within reasonable time limits, as specified by the *model designer*.
4. The modeled system shall be able to reach steady-state, see 7a for example.



(a) Example of simulated QoIs that have succeeded in reaching steady-state. These results correspond to a passed evaluation, marked with a green dot in Figure 9.



(b) Example of simulated QoIs that have failed to reach steady-state. These results correspond to a failed evaluation, marked with a red cross in Figure 9.

Figure 7. Example of simulation results for different test coordinates, see Figure 5 for an overview of the simulated application example.

To evaluate how hulls can be utilized when aggregating multiple models into a larger system, two Functional Mock-up Units (FMUs) are extracted from the application example architecture. These two models are subjected to an identical verification exploration to generate their corresponding hulls. The hypothesis is that the system can only be utilized in the hull created by the intersection of the internal model hulls (x_{model}) for any shared input variable according to

$$x_{system} = \bigcap_{model=0}^{models} x_{model} \tag{1}$$

where x_{system} represent the system OD. Comparison between the model hulls and measurements of selected variables extracted from the system simulation is to provide a basis for conclusions regarding this hypothesis.

Additionally, a reference implementation of a simple *Bayesian search* is developed to evaluate the challenges and possibilities for further research. Although the method will not be used in the current study, some reasoning around the implementation will be included in the Discussion.

5 Results

When the domains initially exhibit a simple hypercube characteristic, it becomes evident from exploring the reference system (see Figure 2) that this representation is insufficient. In investigated model examples, fitting a hypercube to encompass the passed samples will incur a loss of information. It can also be seen in Figure 9 that the quality of the generated hull is also a factor to take into consideration. Where the 'Original hull' has a very low quality in its representation of the domain against the requirement, the 'Low quality hull' increases that quality but there are still a certain amount of verification experiments within the volume that fails. The last area, 'High quality hull', is a very high-quality area where no experiments have failed, but this is at the cost of utilization of the model; in other words, the high quality hull excludes many operating conditions that are identified as 'Passed' .

A simple nearest neighbor metric has been used for creating the different quality hulls. A metric value for each simulation is calculated based on the ratio of how close the simulation is to both failed and successful simulation, mathematically described through

$$sim_{ratio} = f(sim, sims_{fail})/f(sim, sims_{pass}) \tag{2}$$

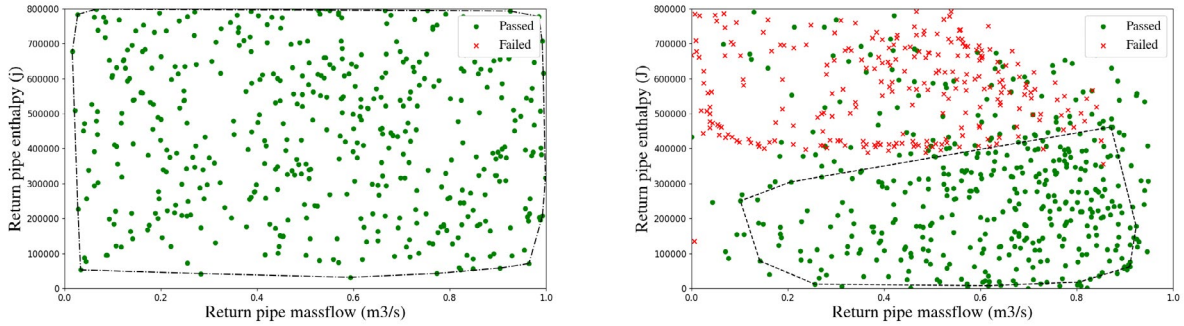
$$f(sim, sims) = \sum_{n=1}^{sims} (1/((distance(sim, sim_n)^2 \tag{3}$$

where sim is the current simulation and $sims_{fail/pass}$ signify all simulations that pass or fail a requirement.

The proposed extension of the SRMD format consists of an *OperationalDomain* tag that encapsulates the OD information. An implementation example is shown in Listing 1, followed by a description of the individual tags and attributes in Table 1, 2, 3, 4 and 5. The included attributes should be viewed as initial examples and should be evaluated further in a broader industrial context to account for requirements from other business domains. A full example can be found in Listing 2 provided in the Appendix.

A note on the types of OD proposed, a hypercube can be expressed as a convex hull but the points needed to do so are 2^n, therefore we also proposed to include "hypercube" as a separate volume type where the geometry is defined by two points "x_{min}, y_{min};x_{max}, y_{max}" for simplicity.

 10.3384/ECP20729 DOI



(a) Results of the consumer model verification and the resulting OD, a total of 300 test are performed.

(b) Results of the *ECS* verification and the resulting OD, a total of 800 tests are performed.

Figure 8. Model ODs that can be used to describe and aggregated system OD.

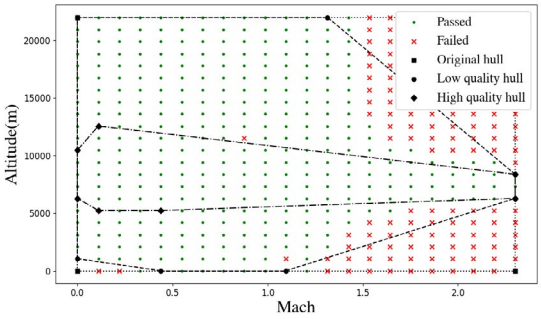


Figure 9. Different quality variants of Operational Domain (OD), represented as convex hulls.

Table 1. OperationalDomain tag details.

Attribute	Description
name	(str) Unique name of the OD
derived	(str) If present, enables traceability to the domain used as a basis for the current exploration
Element	
<Annotations>	If present, FMI or SSP XML standard annotations to support additional information and a human-readable definition of the OD.
<Volume>	Contains the resulting volumetric representation of the OD.
<Requirement>	Defines against what requirements the OD is verified.
<Error>	Provide information regarding the faults within the final hull

Listing 1. SRMD implementation example.

```
<OperationalDomain
    name="domain1.1"
    derived="domain1">
  <Annotations>
    <Annotation type="OD_Information">
      <Info>
        Simulation completed, no errors
      </Info>
    </Annotation>
  </Annotations>

  <Volume
    type="convex_hull"
    points="x1,y1;x2,y2;x3,y3"
    variables="Altitude,Mach"/>
  <Requirement
    simulation_status="no_errors"/>
  <Error fraction="0.1">
      <Point pos="x4,y4">
      ....
  </Error>
</OperationalDomain>
```

Table 2. Volume tag details.

Attribute	Description
type	(enum["convex_hull", "hypercube"]) The volume type OD
points	(array[array[float]]) The coordinates of the points defining the volume.
variables	(array[str]) The variable name mapping towards the model input space.

Table 3. Requirement tag details.

Attribute	Description
simulation_status	(enum["failed", "no_errors", "no_warnings"]) If present, place constraints on simulation execution.
execution_time	(float) If present, place constraints on single-step execution time in seconds.
hull_uncertainty	(float) If present, place constraints on hull uncertainty as the ratio of permissible faults within the volume.

Table 4. Error tag details.

Attribute	Description
fraction	(float) The final ratio between failed/passed points within the hull.
Element	
<Point>	[multiple] If present, coordinates of a failed point within the hull to enable an additional evaluation of credibility.

Table 5. Point tag details.

Attribute	Description
position	(array[float]) Coordinates of a point

5.0.1 OD comparison

When evaluating hulls to establish what performance and usage that can be expected when pairing the application example models, both the consumer and ECS are utilized. A return pipe connecting the consumer and ECS is selected for the comparison and both models were verified using a random search, they both have a nine-dimensional input space where suitable ranges for each input variable were acquired from the respective model designers. The comparison of ODs presented in Figure 8 is conducted over liquid enthalpy and mass flow; these two input variables were selected since they both appear to have some correlation to test failure in both model and system.

Mapping the system usage over the two models, see Figure 11, shows that the hypothesis should not be discarded. The results in this case are promising and the method utilized could give system architects a powerful new tool for evaluating how a system will behave earlier in the product development cycle. However, additional tests involving other models and systems are needed for any strong conclusions to be made but such tests are left for future research.

6 Discussion

O. Balci and Ormsby (2000) argues that a model accreditation recommendation can only be provided for data and scenarios connected to the model's intended use and that any results obtained from conditions outside this scope are not credible. Model exploration is a means to evaluate the model in such a way as to provide evidence supporting that model accreditation can be provided for the full OD. One of the larger questions regarding this is when a requirement can be guaranteed up to a certain confidence level. Taking Figure 9 as an example, the single failed experiment almost in the center of the plot showcases a situation without a simple solution. If the requirement of this particular model dictates a confidence level of 100%, it could be the difference between reusing a model and designing a new one. In some cases, the design cost difference between 100% and 95% can be extensive. Concretizing the confidence level for each requirement enables the creation of a cost-effective solution. As stated Section 2.1, we are to strive towards minimizing implicit requirements. This together with the utilization of a standardized machine-readable format together with the MA standards would fulfill the *Findable*, *Accessible*, *Interoperable*, and *Reusable* principles of FAIR and increase the traceability of model verification.

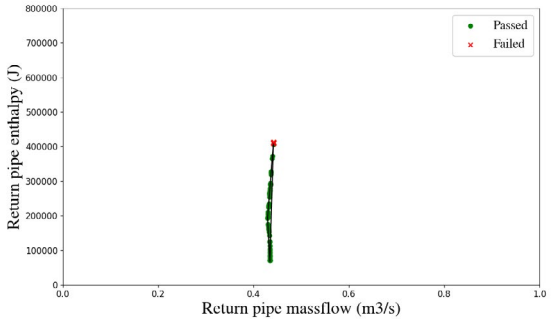


Figure 10. Results of the *system* verification and the resulting OD, a total of 500 tests are performed.

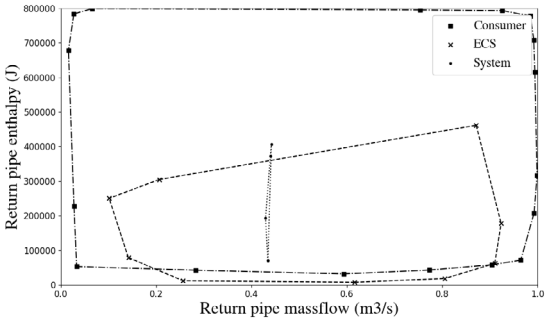


Figure 11. Comparison between the *consumer* model OD, the ECS model OD, and the resulting system utilization OD.

Stating confidence levels in requirements opens up an interesting opportunity for model reuse. As seen in Figure 9, multiple domains for a requirement can be specified depending on the confidence level. The current system may require a 95% confidence level, but during verification, ODs can be created not only for 95% but also for 90% and 80%. This could enable evaluating the model for uses outside the current project that requires a larger OD but has lower requirements regarding confidence level. The same approach may be applied to common requirements such as requirements concerning accuracy and speed. Variations of confidence levels and requirements can, in most cases, be achieved without additional simulations by evaluating already collected data from each simulation against additional requirements. Each model or application area will likely have some specific requirements that may enable easier utilization in new contexts and quick verification in reuse cases. The areas that could benefit the most from this approach will likely become clearer with increased usage of domains in general.

When exploring a model, as established earlier, it is quite ineffective to explore the OD using the described *Grid* or *Random* search approaches. However, as previously stated, a simple *Bayesian search* was implemented to evaluate potential challenges using this method in this context, see Figure 4. It is built upon the assumption more information regarding system behavior can be found in the border regions dividing the passed and failed simulations. It's therefore designed to prioritize new

simulations in such areas. The main finding from this implementation is that a challenge in utilizing a *Bayesian search* as a general solution is in evaluating the choice of tests chosen by the algorithm. For example, evaluating results based on a *Random search* is often straightforward, whereas the corresponding results when using a complex search algorithm may require specialized knowledge of both the search algorithm and the coverage metric. This is especially true as manual evaluation of the exploration strategy becomes more complex with increasing input dimensions. As of now, avoiding specialized search algorithms sidesteps the question of evaluating if the chosen optimization method and *coverage* objective is the correct choice for the current model.

7 Conclusion

Coupling a requirement-driven formulation of a model verification activity in the form of a Operational Domain (OD) would increase traceability and may provide multiple new opportunities for model and system verification. Before simulating, evaluating how aggregated models perform together can provide valuable insights regarding the system's behavior. During simulation, model validity can be monitored and provide warnings for models utilized in an unverified context. After simulating, more computationally intensive verification and debugging can be conducted utilizing logs and results. When combining multiple models, aggregation factors make it difficult to validate the system at scale (Wang et al. 2019), and verification utilized during all phases of usage can be seen as a means to monitor aggregation effects.

Acknowledgements

This research was conducted within the scope of the ITEA4 OpenSCALING project, funded by Vinnova and Saab AB. The authors would also like to thank Dan Louthander for his work with the needed DORIS framework adaptations.

References

Andersson, Henric (2012). "Variability and Customization of Simulator Products : A Product Line Approach in Model Based Systems Engineering". PhD thesis. Linköping University, Machine Design, p. 83. ISBN: 978-91-7519-963-4.

Andersson, Henric and Magnus Carlsson (2012-12-12). *Saab Aeronautics Handbook for Development of Simulation Models. Public Variant*. 1st ed. Linköping University Electronic Press.

Asaeedi, Saeed, Farzad Didehvar, and Ali Mohades (2014). *Alpha-Concave Hull, a Generalization of Convex Hull*. arXiv: 1309.7829 [cs.CG].

Atamturktur, Sezer, Matthew C. Egeberg, et al. (2015-01). "Defining coverage of an operational domain using a modified nearest-neighbor metric". In: *Mechanical Systems and Signal Processing* 50-51, pp. 349–361. DOI: https://doi.org/10.1016/j.ymssp.2014.05.040.

Atamturktur, Sezer, François Hemez, et al. (2009-02-09). "Predictive Maturity of Computer Models Using Functional and Multivariate Output". In: *Proceedings of the 27th Conference and Exposition on Structural Dynamics 2009, IMAC-XXVII*. Orlando, Florida USA.

Balci, O. and W.F. Ormsby (2000-12). "Well-defined intended uses: an explicit requirement for accreditation of modeling and simulation applications". In: *2000 Winter Simulation Conference Proceedings (Cat. No.00CH37165)*. 2000 Winter Simulation Conference Proceedings (Cat. No.00CH37165). Vol. 1, 849–854 vol.1. DOI: 10.1109/WSC.2000.899883. URL: https://ieeexplore.ieee.org/document/899883 (visited on 2024-06-05).

Balci, Osman (1997). "Verification, validation and accreditation of simulation models". In: *Proceedings of the 1997 Winter Simulation Conference*. IEEE Computer Society, Washington, D.C., United States, pp. 135–141. DOI: 10.1109/WSC.1997.640389.

Beisbart, Claus and Nicole J. Saam (2019-04-24). *Computer Simulation Validation: Fundamental Concepts, Methodological Frameworks, and Philosophical Perspectives*. Springer. 1088 pp. ISBN: 978-3-319-70765-5. URL: https://www.ebook.de/de/product/30290750/computer_simulation_validation.html.

Carlsson, Magnus et al. (2012-01). "Methodology for Development and Validation of Multipurpose Simulation Models". In: *50th AIAA Aerospace Sciences Meeting including the New Horizons Forum and Aerospace Exposition*. American Institute of Aeronautics and Astronautics. DOI: 10.2514/6.2012-877.

Coïc, Clément et al. (2021-09). "Modelica, FMI and SSP for LOTAR of Analytical mBSE models: First Implementation and Feedback". In: *Linköping Electronic Conference Proceedings*. Linköping University Electronic Press. DOI: 10.3384/ecp2118149.

DO-178C (2012-01-05). URL: https://www.do178.org/ (visited on 2024-06-03).

Eek, Magnus, Hampus Gavel, and Johan Ölvander (2017-02). "Definition and Implementation of a Method for Uncertainty Aggregation in Component-Based System Simulation Models". In: *Journal of Verification, Validation and Uncertainty Quantification* 2.1. DOI: https://doi.org/10.1115/1.4035716.

Eek, Magnus, Robert Hällqvist, et al. (2016-06). "A Concept for Credibility Assessment of Aircraft System Simulators". In: *AIAA Journal of Aerospace Information Systems* 13.6, pp. 219–233. DOI: https://doi.org/10.2514/1.I010391.

Erdemir, Ahmet et al. (2020-12). "Credible practice of modeling and simulation in healthcare: ten rules from a multidisciplinary perspective". In: *Journal of Translational Medicine* 18.1, p. 369. ISSN: 1479-5876. DOI: 10.1186/s12967-020-02540-4. URL: https://translational-medicine.biomedcentral.com/articles/10.1186/s12967-020-02540-4 (visited on 2024-04-26).

Feurer, Matthias and Frank Hutter (2019). "Hyperparameter Optimization". In: *Automated Machine Learning*. Ed. by Frank Hutter, Lars Kotthoff, and Joaquin Vanschoren. Series Title: The Springer Series on Challenges in Machine Learning. Cham: Springer International Publishing, pp. 3–33. ISBN: 978-3-030-05317-8 978-3-030-05318-5. DOI: 10.1007/978-3-030-05318-5_1. URL: http://link.springer.com/10.1007/978-3-030-05318-5_1 (visited on 2024-06-05).

FMI development group (2019-10-31). *FMI Functional Mock-up Interface*. https://fmi-standard.org/. Accessed: 2019-11-07. URL: https://fmi-standard.org/ (visited on 2019-11-15).

FMI development group (2022-05-10). *Functional Mock-up Interface Specification, FMI for Model Exchange, Co-Simulation, and Scheduled Execution*. Available online: https:

//fmi-standard.org/. Report 3.0. URL: https://fmi-standard.org/.

Fritzson, Peter (2004-01). *Principles of Object Oriented Modeling and Simulation with Modelica 2.1*. Wiley-IEEE Press. ISBN: 9780470545669. DOI: 10.1109/9780470545669.

Gripen Mission Trainers (2024). Start. URL: https://www.saab.com/markets/brazil/press-releases/2023/gripen-mission-trainers-in-Brazil (visited on 2024-05-13).

Hällqvist, Robert (2019-05-14). *Digital Twin for Automated Flight Test Evaluation and Model Validation*. Ed. by VINNOVA, National Aeronautical Research Program 7, Ref Nr. 2019-02760.

Hällqvist, Robert (2023). "On The Realization of Credible Simulations: Efficient and Independent Validation Enabled by Automation". PhD thesis. Linköping University, Division of Fluid and Mechatronic Systems. ISBN: 978-91-7929-597-4.

Hällqvist, Robert, Magnus Eek, et al. (2023-01). "Towards Objective Assessment of Model and Simulator Predictive Capability". In: *Journal of Aerospace and Information Systems (JAIS), AIAA.*, pp. 1–16. DOI: https://doi.org/10.2514/1.I011153.

Hällqvist, Robert, Raghu Chaitanya Munjulury, et al. (2022-09-13). "Realizing Interoperability between MBSE Domains in Aircraft System Development". In: *MDPI: Electronics* 11.18. ISSN: 2079-9292. DOI: 10.3390/electronics11182901. URL: https://www.mdpi.com/2079-9292/11/18/2901.

Hällqvist, Robert, Jörg Schminder, et al. (2018-09-09). "A Novel FMI and TLM-based Desktop Simulator for Detailed Studies of Thermal Pilot Comfort". In: *Proceedings of the 31st Congress of the International Council of the Aeronautical Sciences*. International Council of the Aeronautical Sciences, ICAS20180203. ISBN: 978-3-932182-88-4.

Herlihy, Maurice and Nir Shavit (2012-06-25). *The Art of Multiprocessor Programming, Revised Reprint*. Google-Books-ID: vfvPrSz7R7QC. Elsevier. 537 pp. ISBN: 978-0-12-397795-3.

International Council on Systems Engineering (2015). *Systems Engineering Handbook*. 4th ed. John Wiley and Sons, Inc. ISBN: 9781118999400.

ISO 2533 (1975). URL: https://www.iso.org/standard/7472.html (visited on 2024-05-27).

ISO 26262 (2018). *ISO 26262*. URL: https://www.iso.org/standard/68383.html (visited on 2024-06-07).

ITEA 3 (2019-11-12). *EMBrACE*. https://itea3.org/project/embrace.html. URL: https://itea3.org/project/embrace.html (visited on 2024-01-26).

Level, S. E. T. (2024). *SET Level - The safety of automated driving*. URL: https://setlevel.de (visited on 2024-04-26).

Ljung, Lennart and Torkel Glad (2004). *Modelbygge och Simulering*. Vol. 2. Studentliteratur. ISBN: 91-44-02443-6.

Martin, Robert and Grigori Melnik (2008-02-01). "Tests and Requirements, Requirements and Tests: A Möbius Strip". In: *Software, IEEE* 25, pp. 54–59. DOI: 10.1109/MS.2008.24.

MIL-STD-3022 (2008-01-28).

Modelica Association (2022-10-27). *SSP Traceability Specification. Version 1.0.0-Beta2*. Standard, unreleased.

Modelica Association Project System Structure and Parameterization (2019). *System Structure and Parameterization*. URL: https://ssp-standard.org (visited on 2019-10-23).

Muller, G. (2020-11-01). *Industry-as-Laboratory Applied in Practice: The Boderc Project, Version: 1.3*. Ed. by University of South-Eastern Norway-NISE. Kongsberg, Norway. URL: http://www.gaudisite.nl/ (visited on 2022-12-15).

Murray-Smith, David (2019-02-08). "Some Issues in the Testing of Computer Simulation Models". In: *International Journal of Business and Technology* 5.1, pp. 1–10. ISSN: 2223-8387. DOI: 10.33107/ijbte.2016.5.1.01. URL: https://knowledgecenter.ubt-uni.net/ijbte/vol5/iss1/1.

NASA STD-7009 (2008). Place: Washington DC.

Ochel, Lennart et al. (2019-03-04). "OMSimulator – Integrated FMI and TLM-based Co-simulation with Composite Model Editing and SSP". In: *Proceeding of the 13th International Modelica Conference*, pp. 69–78. DOI: 10.3384/ecp1915769.

Oprea, Alexandra (2022). *On aircraft simulation in conceptual design*. Linköping: Department of Management and Engineering, Linköping University. 1 p. ISBN: 978-91-7929-476-2.

Pokojski, Jerzy, Lech Knap, and Stanisław Skotnicki (2021). "Concept of a Multi-Criteria and Multi-Disciplinary Design Activity Supporting Tool in the Design and Development Process of CPS". In: pp. 113–122. DOI: 10.3233/ATDE210089. URL: https://ebooks.iospress.nl/doi/10.3233/ATDE210089 (visited on 2024-05-01).

Pop, A. and Peter A. Fritzson (2004). "The Modelica Standard Library as an Ontology for Modeling and Simulation of Physical Systems". In: URL: https://api.semanticscholar.org/CorpusID:26864047.

Potts, C. (1993-09). "Software-engineering research revisited". In: *IEEE Software* 10.5, pp. 19–28. DOI: doi:10.1109/52.232392.

Pugh, Ken (2010-12-22). *Lean-Agile Acceptance Test-Driven Development: Better Software Through Collaboration*. Upper Saddle River, NJ. ISBN: 978-0-321-71408-4.

Raymer, Daniel P. (2018). *Aircraft design: a conceptual approach*. 6th ed. American Institute of Aeronautics Astronautics. ISBN: 9781624104909.

Riedmaier, Stefan et al. (2020-08). "Unified Framework and Survey for Model Verification, Validation and Uncertainty Quantification". In: *Archives of Computational Methods in Engineering*. DOI: https://doi.org/10.1007/s11831-020-09473-7.

Robinson, Stewart et al. (2004-11). "Simulation model reuse: definitions, benefits and obstacles". In: *Simulation Modelling Practice and Theory* 12.7-8, pp. 479–494. DOI: doi:10.1016/j.simpat.2003.11.006.

Roy, Christopher J. and William L. Oberkampf (2011-06). "A comprehensive framework for verification, validation, and uncertainty quantification in scientific computing". In: *Computer methods in applied mechanics and engineering* 200.25-28, pp. 2131–2144. DOI: 10.1016/j.cma.2011.03.016.

Sargent, Robert G. (2010). "Verification and validation of simulation models". In: *Proceedings of the 2010 Winter Simulation Conference*, pp. 166–183. DOI: 10.1109/WSC.2010.5679166.

Schminder, Jörg et al. (2018). "Pilot Performance and Heat Stress Assessment Support Using a Cockpit Thermoregulatory Simulation Model". In: *Proceedings of the 31st Congress of the International Council of the Aeronautical Sciences*. International Council of the Aeronautical Sciences, ICAS20180463. ISBN: 978-3-932182-88-4.

Sivard, Gunilla (2001). "A generic information platform for product families". QC 20100812. PhD thesis. KTH, Production Engineering, pp. v, 213.

Steinkellner, Sören (2011). "Aircraft Vehicle Systems Modeling and Simulation under Uncertainty". Licentiate thesis.

Linköping University, Division of Machine Design. ISBN: 9789173931366.
The Modelica Association (2019). *Modelica and the Modelica Association*. https://www.modelica.org/. Accessed: 2018-06-21. URL: https://www.modelica.org/ (visited on 2019-11-15).
Wang, Yanan et al. (2019-03-04). "A survey on VV&A of large-scale simulations". In: DOI: 10.1108/IJCS-01-2019-0004.
Wilkinson, Mark D. et al. (2016-03-15). "The FAIR Guiding Principles for scientific data management and stewardship". In: *Scientific Data* 3.1, p. 160018. ISSN: 2052-4463. DOI: 10.1038/sdata.2016.18. URL: https://www.nature.com/articles/sdata201618 (visited on 2024-04-26).

8 Appendix

Full example of an SRMD hull implementation, Listing 2, showcasing an initial hull provided by *model designer* and then the resulting verified hull. These two hulls correspond to the "original hull" and "high quality hull" in Figure 9.

Listing 2. Full SRMD implementation example.

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<srmd:SimulationResourceMetaData version="1.0.0" name="Simulation meta data"
  generationTool="Manual" generationDateAndTime="2024-02-06T13:21:41Z"
  xmlns:srmd="http://ssp-standard.org/SSPTraceability1/SimulationResourceMetaData"
  xmlns:ssc="http://ssp-standard.org/SSP1/SystemStructureCommon"
  xmlns:stc="http://ssp-standard.org/SSPTraceability1/SSPTraceabilityCommon"
  xmlns:xlink="http://www.w3.org/1999/xlink">

  <OperationalDomain name="initial_domain" >
    <Annotations>
      <Annotation type="OD_Information">
        <Info>
          Initial hypercube provided by model designer Mr/Mrs. X.
        </Info>
      </Annotation>
    </Annotations>

    <Volume
      type="hypercube"
      points="0,0;22000,2.3"
      variables="Altitude,Mach"/>
  </OperationalDomain>

  <OperationalDomain name="no_errors_confidence_1" derived="initial_domain" >
    <Annotations>
      <Annotation type="OD_Information">
        <Info>
          High confidence operational domain of no error requirement,
          confidence level of 100%.
        </Info>
      </Annotation>
    </Annotations>

    <Volume
      type="convex_hull"
      points="10500,0;7000,0;5000,0.1;5000,0.4;6000,2.3;8000,2.3;13000,0.1"
      variables="Altitude,Mach"/>
    <Requirement simulation_status="no_errors"/>
    <Error fraction="0.0">
    </Error>
  </OperationalDomain>
</srmd:SimulationResourceMetaData>
```

 10.3384/ECP20729 DOI