SysML v1 landscape => defines the problem (inconsistent, informal guidelines)

FLOW HEURISTICS FOR FUNCTIONAL MODELLING IN MODEL-BASED SYSTEMS ENGINEERING

Yildirim, Unal (1);
Campean, Felician (2);
Korsunovs, Aleksandr (2);
Doikin, Aleksandr (2)

1: Hubei University of Automotive Technology, China;
2: University of Bradford, United Kingdom

ABSTRACT

Model-Based Systems Engineering (MBSE) is increasingly used across industries for the integrated modelling of complex systems to support model-based development and provide enhanced traceability between requirements and verification and validation of the system. This paper seeks to strengthen the function modelling methodology in MBSE by introducing an approach based on flow heuristics guided by the System State Flow Diagram schema. This provides function representations with an enhanced integrity in MBSE facilitating the solution-agnostic architecture modelling, and supports integrated simulation and function failure reasoning based on MBSE. The approach is illustrated with a case study of an electric bicycle implemented in the MathWorks System Composer environment.

Keywords: Functional modelling, Systems Engineering (SE), Product modelling / models, System State Flow Diagram, System Composer

Contact:
Yildirim, Unal
Hubei University of Automotive Technology
China, People's Republic of
U.yildirim@huat.edu.cn

Cite this article: Yildirim, U., Campean, F., Korsunovs, A., Doikin, A. (2023) 'Flow Heuristics for Functional Modelling in Model-Based Systems Engineering', in *Proceedings of the International Conference on Engineering Design (ICED23),* Bordeaux, France, 24-28 July 2023. DOI:10.1017/pds.2023.190

1 INTRODUCTION

Across industries, Model-based Systems Engineering (MBSE) methods are becoming increasingly common in the management of the complexity of product design and development (PD&D) process in the digital age. While in the early days the benefits of digitisation of requirements and other PD&D documentation were commonly invoked for adoption, the rapid proliferation of variants and features offered with products, requiring faster development and validation, brought prominence to MBSE as an effective approach to manage the model-based design and development of complex systems. MBSE is important not just for the productivity benefits derived from requirements traceability and inter-operability, but also for the facilitation of effective interdisciplinary collaboration in the design process required by the now ubiquitous cyber-physical systems and components.

One of the major barriers with the large-scale adoption of MBSE methods and tools is the shortage of trained MBSE modellers. Achieving the paradigm shift for MBSE adoption requires a mass-scale transfer of the methods and tools to systems design practitioners in industry. This requires not just effective training, but also a significant evolution and transformation of the MBSE methods and tools to become more integrated with the "design thinking" and the ecosystem of methods and tools that are required to accomplish PD&D tasks. The increased digital integration of PD&D activities, with increased reliance on simulation-based validation of designs, offers a great potential for the adoption of the MBSE tools, as demonstrated by many examples (e.g. Zhang et al., 2022). However, consistent deployment of MBSE across multi-disciplinary systems with complex multi-physics behaviours is still a challenge that requires further development and adaption of methods.

The Design Theory and Methodology (DTM) research has evolved over many decades to develop and refine methods that assist designers with diverse "designing" tasks and activities. While drawing out universal principles and methods has benefits for developing designing skills in engineers through education and training, careful attention was paid to how the methods would apply to different domains to ensure consistency and productivity. Bringing some of the important kernels behind the DTM methods has great potential to enhance the effectiveness and applicability of MBSE methods. Functional modelling has been recognised by both DTM and MBSE research communities as being of crucial importance for driving systems engineering modelling and development. Functional modelling has been long discussed in the DTM research with a plethora of methods being developed (see Tomiyama et al., 2009 for a review), and criteria for evaluating and benchmarking function models well established (Summers et al, 2017). Guidance for practitioners have also been developed in the form of heuristics, to improve the consistency of the models developed by practitioners (Otto and Wood, 2001).

In the case of MBSE, the common methodologies for function modelling methodologies revolve around software modelling tools available in Systems Modelling Language - SysML (OMG, 2019), including: activity diagrams, sequence diagrams and state machine diagrams. The most common approach revolves around extraction of functional requirements directly from activity diagrams. The consideration of functions as activities is useful as it enables a broader multidisciplinary engagement in the capture of functions and requirements, directly associated with the consideration of use cases describing the way the users interact with the system of interest. Functions are then allocated to components as design solutions for achieving the functions, thus providing the physical architecture (PA) of the system, without the need for a functional architecture (FA) to be defined. While this provides an effective way to manage the requirements, it often limits the use (e.g. for performance simulation) and re-use (e.g. for product variants and generations) of the model. As a function modelling method, this approach fails several of the criteria defined for the functional models and representations (Summers et al, 2013), in particular consistency. Sequence diagrams and state machine models can provide more consistent models, but their application is limited in relation to systems that have significant physical behaviour.

Significant effort in MBSE research has recently been put towards for the development of better frameworks for functional modelling (Drave et al, 2020), mostly based on using SysML to model functions as operations with functions-as-blocks representations. This has been driven by the need to support: (i) the re-use of the MBSE models for Product Generation Engineering (Albers et al. 2020); (ii) model-based design assurance PD&D activities, such as FMEA (Pearce & Friedenthal, 2013) or safety analysis (Biggs et al, 2018), which are underpinned by the system's FA; and (iii) the integration of MBSE with system performance or physical behaviour simulation in modelling environments such as Simulink, which are based on mathematically rigorous functional blocks, requiring greater

https://doi.org/10.1017/pds.2023.190 Published online by Cambridge University Press

consistency from the MBSE models. There are many good examples from academic and industry practice (e.g. Hao Yang et al., 2021; Forlingieri & Weilkiens, 2022). However, the methodology illustrated in these examples builds on function analysis driven by the known system structure.
There is still a gap in guiding the solution agnostic functional decomposition and architecting in MBSE across multiple levels of abstraction. This paper aims to address this gap by investigating the way of transferring rigorous function modelling approaches developed through DTM research to MBSE. In particular, this paper focuses on representations of functions defined as operations on flows, based on an analogy with the Systems State Flow Diagram-SSFD (Yildirim et al, 2019), and the use of flow heuristics as defined by Otto and Wood (2001) to guide practitioners with developing functional models. The MathWorks System Composer is used as MBSE modelling environment for the case study employed to illustrate the proposed method. The contributions of the paper are as follows:

1. Introduce a methodology for function modelling in MBSE based on function flow heuristics in conjunction with use-case analysis, to support solution agnostic FA, in relation to the level of resolution of the analysis;
2. Illustrate the methodology with a case study of an electric bicycle (e-bike);
3. Discuss directions for further research to improve consistency of function modelling in MBSE.

The paper is organised as follows: Section 2 introduces a review of related literature; Section 3 outlines the research methodology, while Section 4 introduces the demonstration of the case on the e-bike; Section 5 concludes the paper with discussion.

2 REVIEW OF RELATED LITERATURE

2.1 Overview of function modelling methods

The flow-based function modelling methodology of Pahl et al. (2007) has set the basis for various well-established modelling schemes in many engineering disciplines including Stone and Wood (2000) and Ulrich and Eppinger (2003). The crux of this methodology is to decompose the main function of a system into subfunctions along with the definition of associated input and output flows of energy, material and signal (information). There are many variations of this methodology. For example, the Integration DEfinition for Function modelling-IDEF0 (Buede, 2009), extended Pahl's et al. (2007) methodology by complementing the inputs and the outputs flows of material, energy, and information with flows for “controls” and “mechanisms”. Functions can also be represented in terms of state transitions as in the Contact and Channel Approach - C&C^2-A (Matthiesen and Ruckpaul, 2012) where a state is defined when system variables take a constant value. A sequence of at least two states determines function of a system. The definition of a function requires at least two Working Surface Pairs (WSP), Connecting Channel Support structure (CSS), and at least two connectors. A set of WSPs and CSSs constitutes a sequence, and functions of the system can be defined by relating a state to other sequences. Some approaches using the concept of “behaviour” in function modelling (e.g. Object–Process Methodology - OPM of (Dori, 2016)) and also represent functions of a system in terms of state transitions. Eisenbart et al. (2016) introduced an integrated function modelling (IFM) approach, providing a functional model of a system with reference to the views of use case, state, interaction, actor, effect, and process flow.
Yildirim et al. (2017) introduced System State Flow Diagram (SSFD) as a systematic approach for function modelling underpinned by a state-based representation of the flows through a complex system. The flow heuristics of Otto & Wood (2001) have been adapted and enhanced to guide the practitioners in carrying out function analysis and decomposition of complex systems systematically. Campean et al. (2018) discussed the use of the SSFD in nested function modelling within use cases of a system and with consideration of the multi-level architecture of a system.

2.2 Overview of function modelling in MBSE

The use and benefits to industry from MBSE methodologies have been discussed, e.g. by Estefan (2008) and Estefan & Weilkiens (2022), with some examples of successful implementation (Davey, 2022).
SysML (OMG, 2019), as the most prevalent MBSE tool in academia and industry (Albers and Zingel 2013; Lu et al, 2018), provides a variety of pre-defined diagrams to be used in model-based development of multidisciplinary systems. Behaviour diagrams of SysML enable to capture functional requirements of a system across its lifecycle use cases which are represented in a use case diagram showing the

https://doi.org/10.1017/pds.2023.190 Published online by Cambridge University Press

functionality of a system to achieve various goals via actors. State machine diagrams, sequence diagrams and activity diagrams serve to the realization of these use cases. State machine diagrams, originating in Harel's (1987) state charts, provide a representation of states as blocks, with lines denoting transitions between states. While state machine diagrams focus on the states for function modelling of a system, activity diagrams provide a different perspective into the functions of a system by relating activities to the flow of inputs, outputs, and controls. Activity diagrams also support the implementation of Enhanced Functional Flow Block Diagram, commonly used in systems engineering (Weilkiens, 2006). The focus of sequence diagrams is to map message-based information interactions between a system and its actors. While these diagrams are used in the description of the flow of control in terms of messages, they also have the potential of representing the flow of material and energy (Friedenthal et al., 2012).
Numerous scholars intended to improve practical applicability of DTM function modelling approaches by integrating them with SysML. Eisenbart et al. (2015) aimed to provide a basis for this by introducing a comparison between the IFM framework and SysML. Zingel et al. (2012) combined C&C²-A with SysML with the aim of enhancing capabilities of C&C²-A in function modelling of technical systems.
Grobshtein and Dori (2011) focused on creating synergies between SysML and OPM based on their respective strengths and weaknesses by introducing automatic generation of several SysML views from an OPM model. While SysML has been used as a de-facto standard since 2006, OPM has been published as ISO 19450 in 2014 to be used for producing conceptual models at various extents of detail.
Some scholars modified SysML behaviour diagrams at various degrees in order to enhance their capabilities in the capture of functional requirements. Sequence diagrams seem to be the most affected diagram in this case. Zingel et al. (2012) used sequence diagram in the introduction of a test case by representing the flow of energy with respective function requirements articulated in verb+noun form between actors of a system. Zhu et al. (2019) proposes a similar methodology in the extraction of functions from sequence diagrams. Both Zingel et al. (2012) and Zhu et al. (2019) also introduced the development of function-based models from activity diagrams. The Enhanced Sequence Diagram - ESD (Yildirim and Campean, 2020) augments traditional sequence diagrams with flows / exchange-based information. The ESD provides a rigorous and information rich graphical representation as a strong basis for functional requirements extraction from multiple use case analysis.

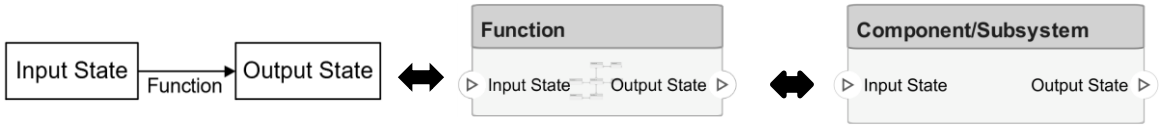
3 RESEARCH METHODOLOGY

The following sections describe our proposed framework for function modelling in MBSE. The framework focuses on a top-down system decomposition and modelling, with implementation illustrated in the MathWorks System Composer (SC). The objective is to develop a function modelling framework to assist engineers with the analysis of a system on function blocks, starting with the higher levels of abstraction, and consideration of function analysis / architecture first, to maintain a solution agnostic model. The intent is also to facilitate the development of executable MBSE models that can be directly simulated at various level of nested architecture development.

3.1 Functions as operations

As discussed in Section 2.1, the state transition / transformation paradigm is common for multiple function modelling approaches. Using the SSFD framework as reference, states are conceptualised in relation to an object with measurable attributes, represented in a box, and function are defined as state transition (see Figure 1.a). Block components can be used in SysML to represent multiple conceptual entities, including functions as well as components (MathWorks, 2022), as shown in Figure 1.b and 1.c, respectively, based on SC. The ports of a function block denote the inputs and outputs to the block. In SC these are specified using stereotypes that include the measurable attributes, similar to the description of states in SSFD. Thus, the SC Function block in Fig 1.b, including an input state and an output state, is a model equivalent to the SSFD representation in Fig 1.a. An allocation in SC establishes a directed relationship from architectural elements (including components and ports) in FA to architectural elements in PA. The concept of "component" in SC allows the allocation of functions to components through ports, as shown in Fig 1.c. where the same input and output states of function in Figure 1.b are allocated to the component/subsystem.
Representing functions as operations on flows supports engineers with the correct capture of the functional requirement, with specification of transformation directly related to the states, i.e. "transform object <input state> to object <output state>", as per the SSFD guidance.

https://doi.org/10.1017/pds.2023.190 Published online by Cambridge University Press



a. SSFD function representation b. SC Function Block c. SC Component Block

Figure 1. SSFD and SC function representation

3.2 Development of function models using flow heuristics

MBSE and DTM approach functional decomposition from different perspectives.

In DTM, function modelling based on operations on flows focuses on describing "the way of achievement" through function chains within a device-centric view (Chandrasekaran and Josephson, 2000). As a way of guiding the practitioner to develop models for complex systems, Otto and Wood (2001) have introduced three flow heuristics: (i) *main* (or *dominant*) *flow* - identifying the functional chain / structure associated with the achievement of the main function of the system; (ii) *connecting flow* - to introduce a flow (as function structure) of additional resources which needs to be "connected" to the main flow; (iii) *branching flow* - introduce bifurcating flows that create parallel function chains. Yildirim et al (2017) have also discussed the conditional fork / join node to denote either connecting or branching flows depending on logical or parametric conditions.

In MBSE, function modelling is driven by the consideration of Use Cases (UC), where each UC describes how an actor (including user) interacts with the system and its context to achieve a desired goal or intent. The common approach is to use an activity-based decomposition to describe the UC. Capturing functions as operations would provide a more consistent function model, and using the *main flow heuristic* provides strong guidance. Focussing on the main UC goal, and using the SSFD logic, the functional requirement can be identified through defining the input and the outputs states. The function decomposition structure based on operations on flows is aided by the SysML behaviour modelling diagrams, in particular the activity diagram or sequence diagram.

The main flow heuristic will be in many cases sufficient to capture the function sequence (as a linear chain) associated with a UC at the first level of decomposition. However, more complex UCs (which involve multiple flows associated with different chains of activities that together deliver the UC goal) or the synthesis of a system function model across multiple UCs, will require the combination of flow sequences corresponding to different UCs (or subcases of more complex UCs). This can be achieved either through the connecting or the branching flow heuristic, often involving logical conditions.

3.3 Abstraction of function modules

A common approach in MBSE is to allocate requirements (including functional requirements), captured from the analysis of activities, directly to Physical Architecture (PA) blocks, identified as the design solution for system. In many cases, multiple requirements / functions are allocated to the same design element. A proper FA is not always defined, given that the design analysis can be completely conducted based on the PA. The analysis at the next level in the system decomposition will start with each PA block / element, looking at how it can be decomposed in terms of logical and physical architecture to deliver all the requirements. While this provides strong traceability, the fundamental problem is that the MBSE models are always solution dependent (describing how the chosen physical architecture delivers the functions). This not only contradicts the stated principle that the function architecture should always be considered first, but it also brings several problems for systems engineering practice. If variants of a product have different allocation of functions to PA, this means that each variant will have its own MBSE model. While in most cases this can be managed by the MBSE software through complex requirements management solutions, it does make interchangeability more difficult to visualise and check. This will have an immediate implication for the PD&D Project Management (PM) task allocations. Depending on the function allocation to PA, some functions and requirements will be the responsibility of one team for Variant A, whereas for Variant B they will be allocated to another team, which will not be an acceptable PD&D PM practice. Not having a proper FA will have further consequences including problems with carrying out simulation based on the MBSE model, and supporting FMEA and safety analysis from the generated MBSE model.

Decomposition of function models through different levels of abstraction is not straightforward in DTM research either. The DSM approach has been widely used for development of modules, but this

https://doi.org/10.1017/pds.2023.190 Published online by Cambridge University Press

implies that PAs need to be considered. This is feasible (Li et al., 2021), but it does require significant concurrent effort on the analysis of the whole manifold of possible solutions.
This paper proposes a much simpler approach, intended to support the representation of the functional models based on the topological analysis of the function model, guided by two main principles:

1. A series of sequence of operations on a flow can be abstracted in a single functional block (Functional Architecture - FA), allocated to a main block in the PA; for longer sequences, multiple blocks can be considered if common functionality can be abstracted; and
2. Connecting or branching flows associated with separate sequences (which could involve series of operations) will normally require a separate design element to perform the operation; in practice, this functional interface element could be associated with a main block in the PA, depending on the architecture choice. However, for the FA representation it is recommended that functional coupling elements are kept as a separate function element.

3.4 Proposed MBSE functional modelling methodology

Figure 2 illustrates the integration of the proposed MBSE function modelling methodology, with the recommended methodological steps indicated on the diagram.

- Step 1: use case analysis and the development of a system use case diagram;
- Step 2: focuses on the development of a functional model based on the flow heuristics. The recommendation is to develop independent functional sequence models (as FAs) for each UC based on the main flow heuristic (as sub-steps, 2.1), followed by the integration into a whole system functional model by using the connecting and branching flow heuristics (sub-step 2.2);
- Step 3: abstraction of the function modules using the principles described above;
- Step 4: development of a Physical Architecture (PA) coupled with the FA; in SC this normally involves the use of the allocation editor (4.1), followed by the representation of the PA (4.2). Given that detailed functional chains are available from UC-guided application of flow heuristics (2.2), the PA can be further decomposed to show the inner structure of the component blocks (4.3).

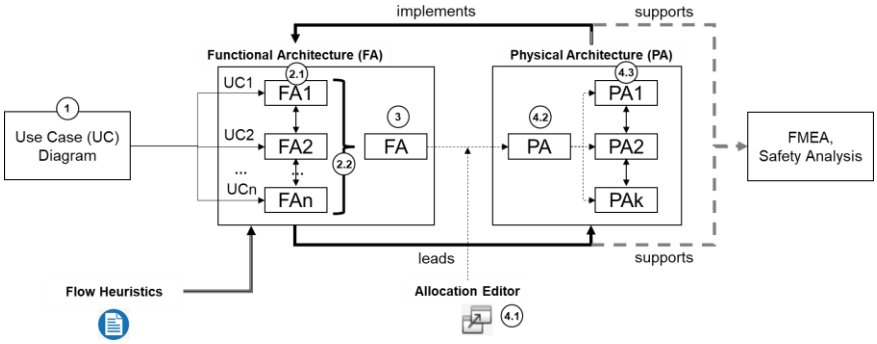


Figure 2. Proposed MBSE functional modelling framework

4 E-BIKE CASE STUDY

Figure 3 illustrates the flow of the methodology applied to an electric bicycle (e-bike) case study.
Step 1 - Use Case (UC) Analysis: For the purpose of this case study we have considered a simplified set of use cases defined from the user perspective (see Figure 3) including: (i) UC1: pedal e-bike, where the e-bike is used like a normal push-bike; (ii) UC2: pedal e-bike with (electric) power assistance; this includes two main activities, associated with the provision of power assist (as additional resource flow for propulsion) by the e-bike system (UC2.1) and the user interaction with the system to request power assistance, i.e. the user control of the feature (UC2.2); and (iii) UC3: charge e-bike.
Step 2 - Develop the Functional Architecture (FA) using Flow Heuristics: In the first instance (Step 2.1 in Figure 3), we consider the development of a FA for each of the UCs, based on the decomposition of activities as function flows via *main flow heuristic*, showing the "way of achievement" as function flows (illustrated by UCs 1, 2.1, 2.2 and 3 in Figure 3). In the following Step 2.2 (Figure 3), a unified FA is assembled by using the other flow heuristics, in particular the *connecting flow heuristic and conditional fork/join node* in this example. This operation is facilitated by the analysis of the states involved in the definition of functions blocks. For example, the final function block of UC 3 refers to the same energy flow as the first function block of UC 2.1, shown in

https://doi.org/10.1017/pds.2023.190 Published online by Cambridge University Press

red frame. This provides a basis for the *connection* of these flows, around the same function block "Store Energy (DC)". It should be noted that this connection is conditioned by the value of attributes, to ensure that charging is not permitted while riding. Similarly, the function block "Convert DC to AC" appears on both the function flows associated with UC 2.1 and UC 2.2, highlighted with blue frame. These two function flows can be combined by using the *connecting flow heuristic*, on the "Convert AC to DC" function block.

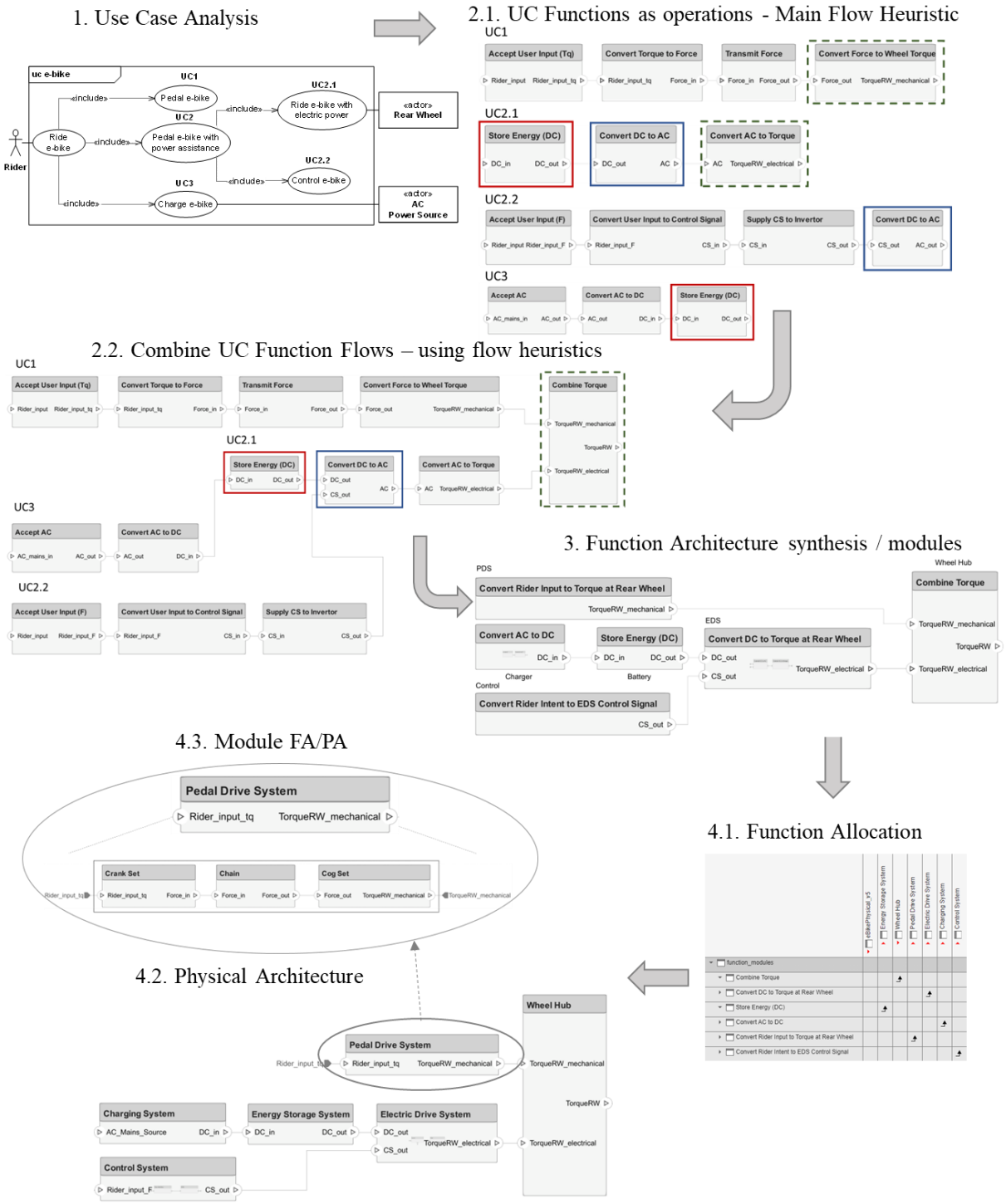


Figure 3. The flow of the methodology, illustrated in Figure 3, based on an e-bike

Finally, in order to deliver the user required functionality of UC2 (pedal e-bike with power assistance), we need to combine the output of final function block of UC1 (Torque_mechanical) with the final function block output of UC2.1 (Torque_electrical) - shown in dotted frame. This requires the introduction of a new functional block to perform the coupling, applying the *connecting flow heuristic*.

https://doi.org/10.1017/pds.2023.190 Published online by Cambridge University Press

This is shown in Figure 3 as "combine torque" function block (this would have been represented by a fork node in a SSFD representation).

Step 3 - Synthesis / Abstraction of Function Architectures: The derived function model includes a detailed analysis of the flows resulting in a complex function structure. This FA model can be further abstracted by defining function modules as higher-level functions, based on the principles outlined in Section 3.3. Specifically, functions that are part of a serial function structure can be conveniently abstracted into a higher-level function, as shown in Figure 3. For example, function blocks of UC 1 are allocated to "Convert Rider Input to Torque at Rear Wheel" function block which takes the input of the first function block and the output of the last function block of UC 1 as the input and the output, respectively. The architecture for this high-level function is articulated in a generic way as "Pedal Drive System (PDS)" in Figure 3. This promotes the consideration of different architecture solutions to the sub-function blocks (preserved as separate function blocks) associated with combinations of flows.

Step 4 - Allocation of FA to Physical Architecture (PA): An allocation in SC establishes a directed relationship between FA elements (including components and ports) and PA elements. In the case of the allocation of multiple FA elements of the e-bike, the allocation editor of SC allows to allocate FA elements to PA elements (Step 4.1). This facilitates consideration of multiple physical architecture options, which may be associated with different technologies. The allocation table in Figure 3 shows the allocation of high-level function blocks defined in Step 3 to conceptually defined physical elements; for example, the Pedal Drive System (PDS) to "Convert Rider Input to Torque at Rear Wheel" (high level) function block. On this basis, in Step 4.2 high-level PA blocks are shown based on these allocations. The allocation editor encompasses the whole FA and PA elements in a way that they can be decomposed into sub-elements. E.g., Step 4.3 shows the PDS, a PA element (high level), can be decomposed into sub-PA elements based on the initial UC 1 function structure.

5 DISCUSSION AND CONCLUSIONS

This paper discussed a methodology for the development of solution agnostic function architectures in MBSE, based on integrating the well-founded approaches from DTM function modelling practice into the MBSE process flow, implemented in the MathWorks System Composer (SC) environment. SC was used as preferred modelling environment because of the broader goal of the research to systematically integrate MBSE multi-level complex systems modelling with the system simulation in Simulink / Simscape to support robustness focussed virtual verification and validation. A robustness focussed verification plan aims to demonstrate that function failure modes are avoided in test cases defined to represent uncertain operating conditions. Therefore, verification planning is necessarily underpinned by function failure reasoning, which in turn is driven by the functional models of the system (Campean et al, 2013). Solution agnostic FAs at a given level of resolution in a system decomposition also supports the generation of re-usable specifications for verification methods, as well as delivering the PD&D benefits discussed in Section 1.

The proposed methodology builds upon emerging trends in SysML to model functions as operations with representation of functions as blocks (e.g. Drave et al, 2020), and introduces a systematic methodology based on the operations on flows heuristics of Otto & Wood (2001).

The e-bike case study has provided a meaningful illustration of the key elements of the methodology. Firstly, the independent modelling of use cases with function structures based on the main flow heuristic is useful as it supports the analyst to capture the model for each use case. This might require the decomposition of the UC into sub-cases e.g. based on major activities, as illustrated by the "ride e-bike with power assistance" UC. The development of function sequences ensures the consistency (or mathematical correctness as a first order logic model) of the functional model and the associated requirements captured, essential if the execution of the model is desired. By contrast, the common practice of direct mapping of requirements to the PA makes the subsequent derivation of function models more difficult and often inconsistent with the executable model purpose.

The flow heuristics were then used to combine the function structures developed for the individual UCs. The case study has illustrated the value of the identification of flows and input/output states, as this has provided a basis for connecting function flows, either through concatenation (UC3 + UC2.1) or connecting different flows (UC2.1 + UC2.2) for achieving a function. We have also illustrated the use of the connecting flow heuristic via a new function to combine energy flows (UC1 + UC2.1). This

https://doi.org/10.1017/pds.2023.190 Published online by Cambridge University Press

is not an exhaustive list, and further research to establish a complete set of methods for implementing the flow heuristics is still needed.
The abstraction of function structures into simpler function modules has been also illustrated, providing a compact function model, that can be argued that contains the sufficient set of functions for delivering the functionality of the system across the set of use cases considered. Demonstrating the invariance of this FA model in relation to the architecture choices at lower levels of resolution requires further consideration of other UCs and features of the e-bike.
The allocation of FA and PA in SC was also illustrated in Figure 3, also showing the nested decomposition of FA / PA into subsystems. As the subsystem models are further defined, the SC model of the e-bike could be executed in Simulink or Simscape, with parameters representative of the different UC scenarios, towards the robustness verification of the system. Development of further features or variants of the system can be easily added, fast-tracking the verification and validation of systems. This suggests that the proposed methodology can leverage automation and computation to support simulation of the system dynamic behaviour by providing a thread of data flow from use cases to physical architecture on the basis of the definition of measurable parameters on the functional model. This provides a foundation for the evolution of the developed MBSE based model into a comprehensive model-based environment incorporating robustness focussed verification and validation.
Further work will focus on demonstrating the integration with simulation modelling for robustness testing, and the further validation of the proposed methodology, in particular in relation to scalability.

ACKNOWLEDGMENTS

The research presented in this paper was partly supported by the Doctoral Research Start-up Fund of Hubei University of Automotive Technology, grant number BK202204. Their support is gratefully acknowledged.

REFERENCES

Albers A and Zingel C (2013) Challenges of model-based systems engineering: a study towards unified term understanding and the state of usage of SysML. In: Proceedings sf Smart product engineering: the 23th CIRP design conference. Bochum, Germany. pp 83–92. https://doi.org/10.1007/978-3-642-30817-8_9

Albers, A., Fahl, J., Hirschter, T., Haag, Hünemeyer, S. and Staiger, T. (2020) Defining, Formulating and Modeling Product Functions in the Early Phase in the Model of PGE — Product Generation Engineering. IEEE International Symposium on Systems Engineering (ISSE), pp. 1-10, https://dx.doi.org/10.1109/ISSE49799.2020.9272222.

Biggs, G., Juknevicius, T., Armonas, A. and Post, K. (2018) Integrating Safety and Reliability Analysis into MBSE: overview of the new proposed OMG standard. INCOSE International Symposium, 28: 1322-1336. https://doi.org/10.1002/j.2334-5837.2018.00551.x

Buede, D.M. (2009) The engineering design of systems: models and methods. 2nd ed., Wiley series in systems engineering and management. John Wiley & Sons, Hoboken, N.J. https://dx.doi.org/10.1002/9780470413791

Campean, F., Henshall, E., and Rutter, B. (2013) Systems Engineering Excellence Through Design: An Integrated Approach Based on Failure Mode Avoidance. SAE Int. J. Mater. Manf. 6(3):389-401, https://doi.org/10.4271/2013-01-0595.

Campean, F., Yildirim, U. and Henshall, E. (2018) Synthesis of functional models from use cases using the system state flow diagram: a nested systems approach. Paper presented at the 15th International Design Conference, Dubrovnik, Croatia 21-24 May 2018, 2833-2844. 10.21278/idc.2018.0543.

Chandrasekaran, B. and Josephson, J.R. (2000) Function in device representation. Engineering with Computers, 16, pp. 162-177, https://doi.org/10.1007/s003660070003.

Davey, C., 2022, Ford's Connected-Agile, Model Based Systems Engineering and Simulation Journey….so far, 32nd Annual INCOSE International Symposium, Detroit, MI, USA, June 25-30.

Dori, D. (2016). Model-Based Systems Engineering with OPM and SysML. Springer New York, New York, NY. https://doi.org/10.1007/978-1-4939-3295-5

Drave, I. et al. (2020) Modeling mechanical functional architectures in SysML. In Proc 23rd ACM/IEEE Int Conf MODELS '20, 79–89. https://doi.org/10.1145/3365438.3410938

Eisenbart, B., Mandel, C., Gericke, K. and Blessing, L. (2015) Integrated function modelling: comparing the IFM framework with SysML. Proc 20th ICED15. Milan, 27-30 Jul

Eisenbart, B., Gericke, K., Blessing, L.T.M. et al. A DSM-based framework for integrated function modelling: concept, application and evaluation. Res Eng Design 28, 25–51 (2017). https://doi.org/10.1007/s00163-016-0228-1

Estefan, J. (2008) INCOSE Survey of MBSE Methodologies. INCOSE TD 2007-003-02, Seattle, WA, USA.
Estefan, J.A., Weilkiens, T. (2022) MBSE Methodologies. In: Madni, A.M., Augustine, N., Sievers, M. (eds) Handbook of Model-Based Systems Engineering. Springer, https://dx.doi.org/10.1007/978-3-030-27486-3_12-1
Forlingieri, M. and Weilkiens, T. (2022), Two Variant Modeling Methods for MBPLE at Airbus. INCOSE International Symposium, 32: 1097-1113. https://doi.org/10.1002/iis2.12984
Friedenthal, S., Moore, A. and Steiner, R. (2012) A practical guide to SysML: The Systems Modeling Language. 2nd edition. Morgan Kaufmann.
Grobshtein, Y. and Dori, D. (2011), Generating SysML views from an OPM model: Design and evaluation. Syst. Engin., 14: 327-340. https://doi.org/10.1002/sys.20181
Harel, D. (1987) Statecharts: a visual formalism for complex systems. Science of Computer Programming, 8(3), pp. 231-274. https://doi.org/10.1016/0167-6423(87)90035-9.
Hao Yang et al 2021 J. Phys.: Conf. Ser. 1827 012096, https://dx.doi.org/10.1088/1742-6596/1827/1/012096.
Hoffmann, H.-P. (2011) Systems Engineering Best Practices with the Rational Solution for Systems and Software Engineering - Deskbook Release 4.1: Model-Based Systems Engineering with Rational Rhapsody and Rational Harmony for Systems Engineering.
INCOSE. 2010. INCOSE Systems Engineering Handbook. v.3.2,Seattle, WA, USA.
Li, Y., Ni, Y., Zhang, N. et al. (2021) Modularization for the complex product considering the design change requirements. Res Eng Design 32, 507–522. https://doi.org/10.1007/s00163-021-00369-6
Lu, J., Wen, Y., Liu, Q., Gürdür, D. and Törngren, M. (2018), MBSE Applicability Analysis in Chinese Industry. INCOSE International Symposium, 28: 1037-1051. https://doi.org/10.1002/j.2334-5837.2018.00532.x
MathWorks (2022). System Composer, User Guide (R2022b). https://uk.mathworks.com/help/pdf_doc/systemcomposer/index.html
Matthiesen, S., Ruckpaul, A., 2012. New Insights on the contact&channel-approach – modelling of systems with several logical states. International Design Conference pp. 1019–1028.
OMG (2019) OMG Systems Modeling Language (OMG SysML™) Specification Version 1.6. https://sysml.org/.res/docs/specs/OMGSysML-v1.6-19-11-01.pdf
Otto, K. and Wood, K. (2001) Product design: techniques in reverse engineering and new product development. New Jersey: Prentice Hall.
Pahl G., Beitz W., Feldhusen J. and Grote KH. (2007) Engineering design: a systematic approach, 3rd edn. Springer, London, https://doi.org/10.1007/978-1-84628-319-2
Pearce. P. and Friedenthal, S. (2013) A Practical Approach For Modelling Submarine Subsystem Architecture In SysML. Submarine Institute of Australia Science, Technology & Engineering Conference. pp. 347-360.
Stone, R.B., & Wood, K.L. (2000) Development of a Functional Basis for design. Journal of Mechanical Design, 122, pp. 359-370. 10.1115/1.1289637
Summers, J.D., Eckert, C., Goel, A.K. (2017) Function in engineering: Benchmarking representations and models. AIEDAM 31, 401–412. https://doi.org/10.1017/S0890060417000476
Tomiyama, T., Gu, P., Jin, Y., Lutters, D., Kind, Ch., Kimura, F. (2009) Design methodologies: Industrial and educational applications. CIRP Annals 58, 543–565. https://doi.org/10.1016/j.cirp.2009.09.003
Ulrich, K.T. & Eppinger, S.D. (2003) Product design and development. 3rd edn. New York: McGraw-Hill/Irwin.
Weilkiens, T. (2006) Systems Engineering with SysML/UML: Modelling, Analysis, Design. Morgan Kaufmann.
Yildirim, U., Campean, F., and Williams, H. (2017) Function modeling using the system state flow diagram. AI-EDAM, 31:4:413-435. https://doi.org/10.1017/S0890060417000294
Yildirim, U. and Campean, F. (2020) Functional modelling of complex multi-disciplinary systems using the enhanced sequence diagram. Res Eng Design 31, 429–448. https://doi.org/10.1007/s00163-020-00343-8
Zhang, Y.; Roeder, J.; Jacobs, G.; Berroth, J.; Hoepfner, G. (2022) Virtual Testing Workflows Based on the Function-Oriented System Architecture in SysML: A Case Study in Wind Turbine Systems. Wind, 2, 599–616. https://doi.org/10.3390/wind2030032
Zhu, S., Tang, J., Gauthier, J-M, Faudou, R. (2019) A formal approach using SysML for capturing functional requirements in avionics domain. Chinese Jrnl Aero, 32:12:2717-2726, https://dx.doi.org/10.1016/j.cja.2019.03.037.
Zingel, C., Albers, A., Matthiesen, M. and Maletz, M. (2012) Experiences and advancements from one year of explorative application of an integrated modelbased development technique using C&C²-A in SysML. International Journal of Computer Science (IJSC), 39 (2), pp.165-181.

https://doi.org/10.1017/pds.2023.190 Published online by Cambridge University Press