NORDDESIGN CONFERENCE
REYKJAVIK, ICELAND, 12-14 AUGUST 2024

Utilization of SysML System Models for Smart Assembly using Digital Twins

Fabian Wilking[1], Simon Kaup[1], Michel Fett[2], Stefan Goetz[1], Eckhard Kirchner[2], Sandro Wartzack[1]

[1]Friedrich-Alexander-Universität Erlangen-Nürnberg, Engineering Design, Germany
[2]Technical University Darmstadt, Institute for Product Development and Machine Elements, Germany

Abstract: Digital Twins are a promising technology to increase the value of products and create new business models. As models are an essential component of these Digital Twin concepts, the reuse of models is necessary. With this contribution, a methodical approach for the integration of a manufacturing structure into a system model is introduced for early verification activities. Furthermore, a novel utilization concept for transforming SysML models into discrete event models is presented to enable assembly simulations. In addition, it is embedded into a use case scenario for a Digital Assembly Twin.

Keywords: Systems Engineering, Assembly, SysML, Model-Based Systems Engineering, Digital Twin

1 Introduction

Modern technical systems are shaped by an enormous amount of available data. With new technologies like Digital Twins, this available data is used to enrich the value of systems and offers better condition monitoring or new business models (Fett et al., 2023; Schleich et al., 2017). Application scenarios for Digital Twins are manifold and can affect the whole life cycle of a product, including its development and production (Czwick et al., 2020; Stark et al., 2020). For manufacturing and assembly, several concepts were introduced to utilize Digital Twins e.g., creating smart assembly lines for identifying mating parts on the assemblies (Rezaei Aderiani et al., 2021; Wärmefjord et al., 2020; Zheng et al., 2019). Especially for complex products a thorough planning and simulation of assembly activities is necessary to reduce possible changes at late stages of the development or even during production. (Wu and Wysk, 1989).

With Model-Based Systems Engineering (MBSE) a concept is introduced that emphasizes the idea of developing systems with centred models as an authoritative source e.g., for further design decisions (Kruse and Blackburn, 2019). This supports the idea of integrating relevant requirements regarding the production or other stages of the lifecycle early into the models and simulate them. While MBSE is not restricted towards the use of specific models or model languages, the System Modelling Language (SysML) is a broadly accepted modelling language within MBSE to improve communication or document the system architecture (Hick et al., 2019). However, the benefit of MBSE and key characteristics for this concept are still under discussion (Campo et al., 2023; Henderson and Salado, 2021). Many of the vowed benefits can only be quantified to a limited extent. Therefore, it is crucial to enlarge the possible application scenarios for using such system models and increase their value within industrial application. One concept for this, introduced by Wilking et al. (2024), is the utilization of system models, which defines a reuse of system models for multiple purposes e.g., operating and validating Digital Twins (Wagner et al., 2023; Wilking et al., 2022). While system models contain the architecture and behaviour of a system, this information can significantly influence later activities regarding manufacturing and assembly of a system. Hence, it is necessary to introduce early verification and validation mechanisms to avoid later changes caused by leaving out the relevant aspects regarding manufacturing in the system architecture design. An example for this is a discrete event simulation to check, whether existing production capabilities are sufficient to handle the developed system. This is one aspect of early assembly verification, which also includes other aspects like joinability or the identification of assembly partners. This publication focuses on the verification of production capabilities. The usability to model discrete processes within SysML was proven within several publications (Liu et al., 2014; Schoenherr et al., 2014; Schonherr and Rose, 2009). With this technical possibility of modelling, it must be investigated, whether this can be integrated into a utilization framework and is able to provide support for the development of complex technical systems. Thus, evaluation of the reuse and a possible transfer into simulation tools is required. In addition, contributing to a full utilization of models, an enlargement of Digital Twins for assembly activities must be tested to close the loop between the early simulation, results from the assembly process and improvements for the model. This leads to two research questions of this contribution:

1. In which way can an early assembly verification regarding the production capabilities be conducted by utilizing existing SysML models e.g., through model transformation into simulation tools?
2. How can this concept be enlarged towards a Digital Assembly Twin for automated feedback loops that adapt or indicate problems within the system model?

To answer the derived research questions, this contribution is divided into two main sections. First, the enrichment of system models is investigated, to provide additional information for the assembly process. This includes an integration of

these enriched models into a Requirements, Functional, Logical, Physical (RFLP) approach. Second, based on the insights of this methodical integration, an enlargement towards a full Digital Twin is presented as a concept.

2 Related Work

2.1 System Model Utilization

System models serve three purposes for their use and application (Friedenthal et al., 2021). First, improving communication as the used languages, such as SysML, aim the use of transdisciplinary language (Friedenthal et al., 2021). Second, the documentation of system architectures e.g., for legal aspects and traceability. Third, the analysis of the system towards the chosen system architecture. Shortly after the introduction of the SysML, further approaches were published that show the reuse of system models. This reuse can be separated in reuse in terms of recycling that aims the reuse of models or elements of a model e.g., throughout product generations (Albers et al., 2015). Besides that, reuse of system models can be conducted in terms of Utilization (Wilking et al., 2024). The goal is the enlargement of the application space for these models. This is achieved through automated vertical and horizontal distribution of their included information as well as interfacing domain specific tools, such as a model transformation towards assembly simulation. Utilization creates new objectives for initial models and specifically adapted models. As shown in Fig. 1, the Utilization can have different classes to describe the application.

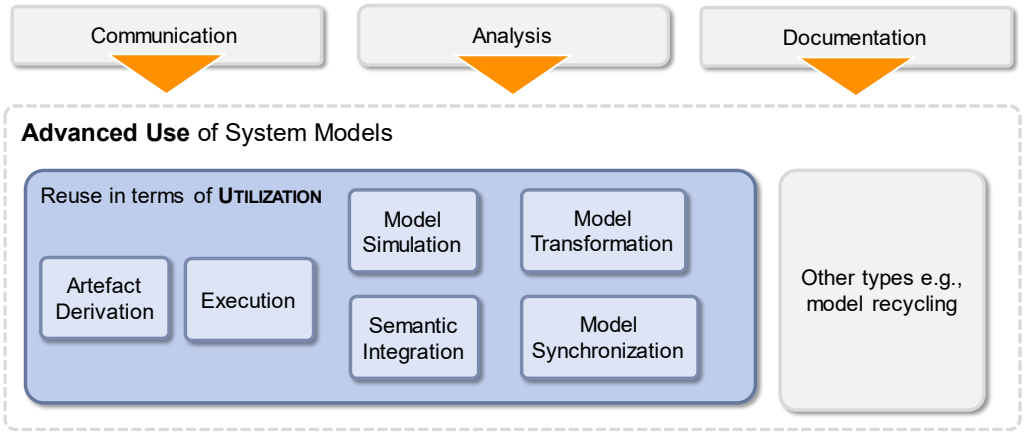

Figure 1: Definition of the reuse in terms of system model UTILIZATION (Wilking et al., 2024)

While MBSE aims the contrast of document-based approaches, documents are still an essential aspect of system development. Especially for documentation purposes and legal aspects these artefacts serve a crucial purpose. Parallel activities to create these artefacts impede the justification of MBSE approaches. Hence, the automated derivation of these documents from existing system models seems to be a convenient automation. Examples for this are the automatic creation of Failure Modes and Effects Analyses or Fault Trees (Girard et al., 2020). In addition to these artefacts, system models are also suitable for defining the behaviour for a system. This can be transformed into executable code and supports the connection between software code and the overall system architecture e.g., for the operation of Digital Twins (Ciccozzi et al., 2019; Wilking et al., 2022). To check these models, model simulation is a common approach in research. These simulations aim the verification or consistency check (Morkevicius and Jankevicius, 2015).

To support the successful implementation of MBSE, system models must be connected with discipline- and domain-specific models to ensure the transfer of the system architecture into the component design, a model transformation. Hence, a consistent tool is mandatory, such as examples show to connect CAD and system models or control simulation in Simulink (Palachi et al., 2013; Schumacher and Inkermann, 2022). In cases where solely partial information is needed, a semantic integration can be applied to extract information from a system model and enrich other models with it (Brahmi et al., 2021). Furthermore, to ensure an authoritative source of truth, bidirectional model synchronization allows the aforementioned model transformation and adds the enrichment of the system model with more detailed information from the domain-specific model. This could be dimensions, test results, etc and must be used cautiously to prevent an information flood within the system model (Giese et al., 2009; Hick et al., 2023).

2.2 Digital Twins for Smart Manufacturing and Assembly

Digital Twins have several different application scenarios and business cases, from very early concepts to improve prototyping to product-service systems. A common vision is the digitalization of manufacturing, coming along with the term industry 4.0 or even 5.0 to increase efficiency and flexibility within manufacturing by connecting value-added chains. Hence, these production facilities contain several different Digital Twins being aggregated to increase the overall

efficiency of manufacturing or lower the cost. The influences on the production stage are manifold. For example, Schleich et al. (2018) show the next generation of geometry assurance by using a Digital Twin. With the scanning, sorting and pairwise matching, fitting assembly partners can be identified to increase manufacturing efficiency. Other examples within manufacturing are the usage of Big Data and learning algorithms to identify new dependencies between development and production (Qi and Tao, 2018) as well as human robot collaborations (Malik and Bilberg, 2018) or tracing product fault sources (Lu et al., 2020).

3 Enrichment of System Models with Manufacturing Information and Model Transformation

A crucial aspect to conduct early verification and validation (V&V) activities within the system architecture, is to reduce deviations or adaptations from the development approach, as they are accompanied by additional effort. A classical approach is the V-Model for developing systems with the left arm focusing on requirements, functions, logical and physical architecture (RFLP). The RFLP approach can be seen as an example to build up the system architecture. The use of other approaches like SYSMOD, OOSEM, etc. is possible, too. Within these phases of the V-Model the system architecture is set up. Thus, some of the elements might already contain relevant information for the manufacturing process. This requires the integration of views into the system model, which is a common scenario within the use of system models (Friedenthal et al., 2021). With these views, relevant information regarding the manufacturing process can be derived. In a common system development process this information would be further used within the domain specific models for the detailed design of the components. At a later stage this would define the manufacturing structure, enabling the engineer to simulate production and assembly processes. However, obstacles at this stage might lead to a change of the system architecture and an iteration of the development process. Therefore, early verification regarding manufacturing seems reasonable. The subsection 3.1 shows the relevant integration into a system model and how the necessary information regarding assembly aspects can be implemented. Based on this, subsection 3.2 shows the tool implementation by directly transforming a SysML diagram into a discrete event simulation model and extracting further information and constraints from other diagrams.

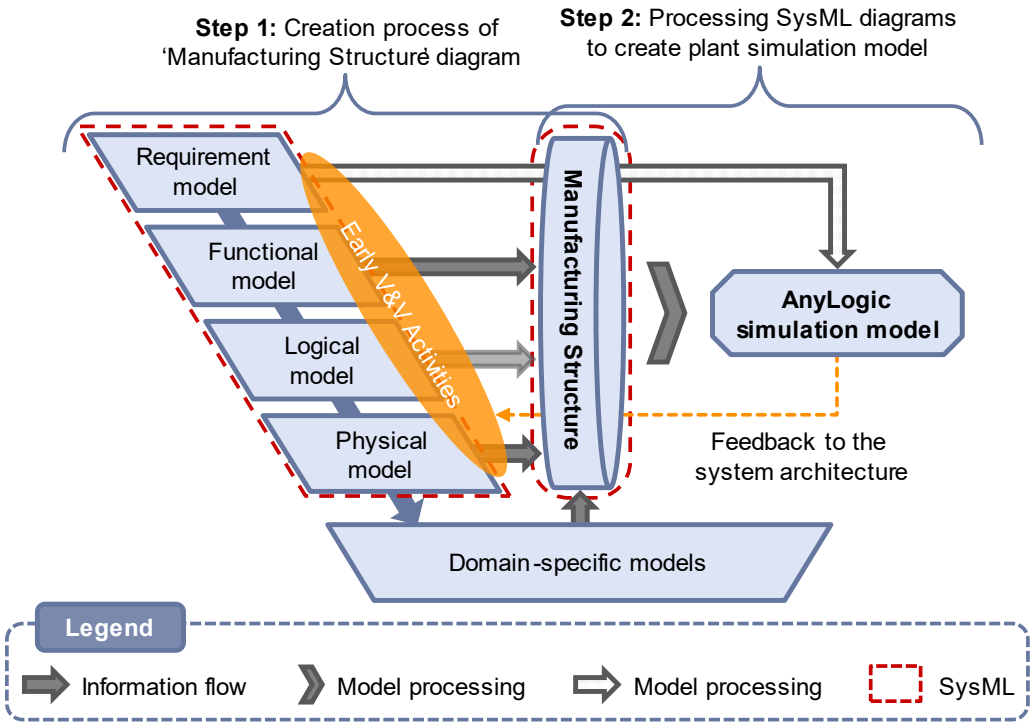

Figure 2: Utilization of System Models containing Information from RFLP towards the Manufacturing Structure

3.1 Integration into RFLP Process and Derivation of Manufacturing Information

For this, a two-step approach is introduced (s. Figure 2). To enable the automatic generation of a plant simulation model it is mandatory to clarify, which information can be used from an initial system model that is created during the system architecture definition. This data must be modelled in a separate diagram and enriched with additional information that is not contained in the existing system model but is required for the creation of the simulation model. In particular, the domain specific models need to be exploited to deduce detailed information demanded by an assembly process. Then, in a second step, this created diagram – called 'Manufacturing Structure' in **Error! Reference source not found.** – must be

transformed into a corresponding simulation model, for which in this case the tool AnyLogic was used. Furthermore, additional information needs to be extracted from the requirement model to complete the simulation model.

The basic development methodology, by which the SysML model of a product or system under development is built, is based on the RFLP approach as part of a V-Model. In context of an implementation the focus is on how a system model can contribute to the generation of a model of its assembly process in a plant simulation software. The starting point is the requirements model in which all essential requirements of the stakeholders are collected. This model can be supplemented by specific requirements that support the assembly simulation. For example, the quantity of a product required by the customer per specific period can be stored. In addition, boundary conditions such as the available number of employees and machines or workstations are defined in this diagram. In addition, the runtime of the machines, i.e. production time per defined time-period, are included. The functional model level addresses the subject, which is being developed. The design and the flow of an assembly process can be influenced by the existence of functions. Thus, associated components can be derived from functions to ensure that they are fulfilled. At this level functions are considered to have an impact on the physical design of a product or system and must be included in the planning of the assembly process. The logical and physical model levels are not sharply separated and layered at this point, since the former has only a minor influence on the design of an assembly process. This comprises the modelling of the operating structure of a system. By an analysis of the logical and physical models one can receive direct information about components of the system. Hence, an initial statement can be posed about the individual assembly steps which will be modelled in the Manufacturing Structure diagram. A further involvement from the domain-specific design to the SysML model is necessary, since this is the only way the detailed assembly processes can be deduced. In the view of the domain-specific models, such as CAD models, a product is detailed down to the component level. However, this is essential to determine the entire assembly sequence regarding a first run through the simulation. Known and reused components will directly make such information available.

The Manufacturing Structure diagram represents the central link between the SysML-model of a product or system on the one hand and the plant simulation model on the other hand. This diagram and its associated diagrams are encapsulated in a separate package ‘Manufacturing Package’ within the package ‘Logical/Physical Package’. The reason for this is, besides the data encapsulation, the simplified processing of the diagrams during the model export and the transformation into the plant simulation model. The number of required process stations can be inferred from the system model. Furthermore, the domain-specific model provides information on the sequence of the process steps as well as on the required processing time of the individual stations. The Manufacturing Structure diagram is based on a Block Definition Diagram. For this example, an angle grinder was used as a showcase for the model transformation and assembly simulation as shown in Figure 3. The diagram contains four blocks representing the process stations and seven further blocks assigning the components to be assembled to the individual stations. The blocks of these components are not created from scratch but are taken from the packages previously developed during the system development process and linked to these, as well as from libraries which contain machinery information for the enterprise. Information about the processing time of a process station is indispensable for further processing. This must be assigned to a block with ‘Manuf. Time’ as a value property. In addition, a time unit must be associated to the corresponding value. If no resources are used, the order in which the process stations are run through is irrelevant for the blank simulation results. Regarding the angle grinder, this means that it is not relevant, whether a block e.g., ‘MountPOU&GU’ followed by the block ‘MountPU&Housing’ is modelled in the simulation software or vice versa. However, if resources are available, the order may affect the simulation results. For this reason, an additional property is assigned to the blocks. The keyword ‘ProcessStation’ defines the position of the block within the process chain and the specified sequence of the process stations reflects the priority according to which the individual stations are processed. Thus, if only one worker is available for three stations, one product is fully assembled before a new one is started. If no sequence is specified, the simulation tool uses default settings leading to significantly higher processing times. Another block that is modelled independently of the assembly blocks in the Manufacturing Structure diagram is the ‘Inspection’ block, which adds an inspection cycle to the simulation model to check the quality of a manufactured product. In addition to its position in the process chain, two probabilities with the keywords ‘Probability of Rejects 1’ and ‘Probability of Rejects 2’ are given. They determine the probability with which a completed device is free of defects and can be mended. The value types of the manufacturing structure, ‘TimeMin’ and ‘TimeS’ and the units ‘Minute” and ‘Second” are defined in a separate diagram, containing all value types and units of the system model.

3.2 Tool Implementation and Model Transformation

While the manufacturing structure described in 3.1 is only represented by a view on a model part and integrated into a modelling process, a significant contribution of this work is the linking between the tools and the transformation of the models. The relevant information of the manufacturing structure and requirements must be extracted to create an executable discrete event simulation model. To achieve this the XMI files of the manufacturing package were exported. These files serve as input for a developed software program performing a transformation by converting relevant information of the relevant diagrams into an XML model file for the simulation tool. Hence, conducting a model transformation in terms of system model utilization (Wilking et al., 2024). Subsequently, this file can be opened and run in the simulation environment to simulate the created assembly line as shown in Figure 3.

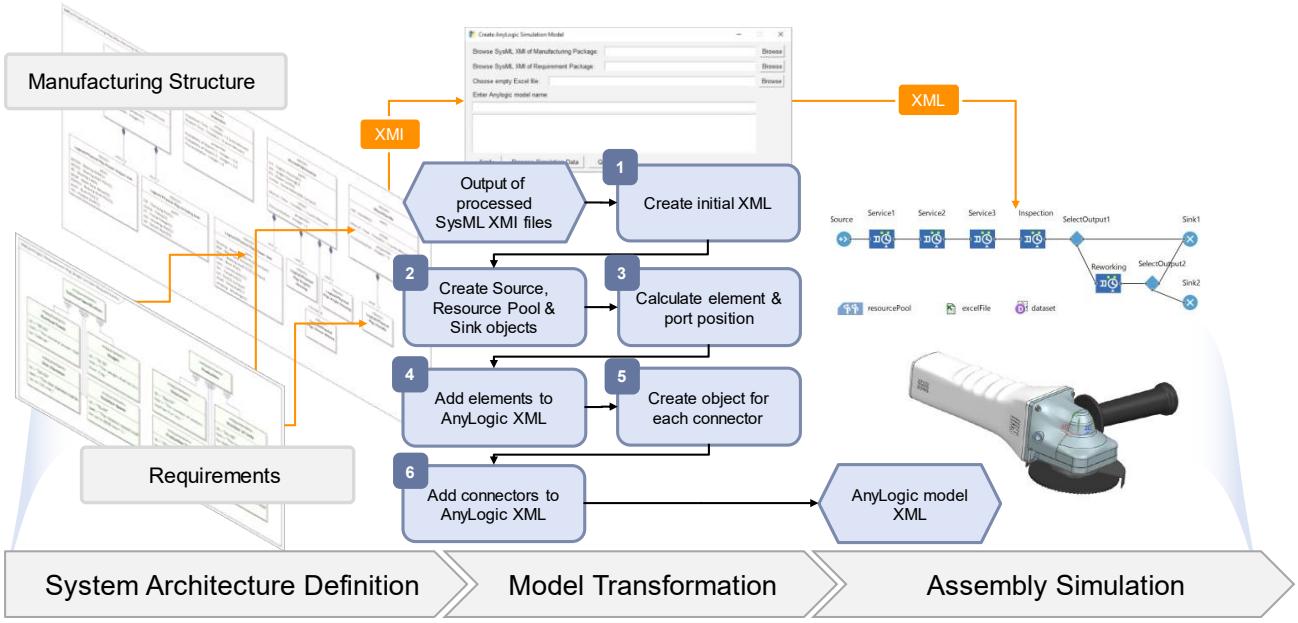

Figure 3: Process for Manufacturing Structure Model Transformation towards AnyLogic Assembly Simulation Model

An essential step on the way to creating a simulation model is the processing and evaluation of the requirement and the specific Manufacturing Structure diagram. The packages containing these diagrams must first be exported into the XMI format. **Error! Reference source not found.** shows the transformation approach for the manufacturing diagram into the discrete event simulation model as well as the necessary steps for the extraction of the requirements. This is the beginning node for the model transformation in Figure 3. With the help of a developed parser, it is possible to search the exported files for the required information and to extract these for further processing. Initially, the XMI file of the Manufacturing Package must be loaded and read, to ensure access to the content of the elements or nodes. The first step is to loop through all child nodes of the <uml:Model> node named <packagedElement> and check their xmi:type attribute for the string 'UML-Class'. This allows specifically to filter only those elements that represent one of the blocks modelled in the Manufacturing Package. In the context of the developed software to transfer the SysML model into AnyLogic, it is relevant to the generation of the discrete event simulation model elements to define the occurrence order of the elements within the process chain. Hence, the previously extracted nodes must be sorted according to their attribute value 'ProcessStation' by analyzing its child nodes. Finally, the sorted nodes are traversed one by one to check the attribute's name of their child nodes for keywords such as 'Manuf. Time' or 'Probability of Rejects' and extract its value. The information is stored by creating a corresponding Python object 'Service' for all nodes found in the first step as well as a Python object 'SelectOutput' for each attribute name to which the string 'Probability of Rejects' is assigned. This is the third and last step (s. **Error! Reference source not found.**) for the SysML manufacturing structure extraction. When parsing the requirement diagram, a requirement to be extracted is first passed as input. At the beginning of this function, the XMI file of the requirement package must be loaded and read. All modeled requirements are subordinated to the <uml:Model> node as child nodes <packagedElement>. Therefore, in a first step, these nodes are searched for the passed requirement. In this case, the name attribute must be checked for equality with the input requirement. The text associated with the given requirement name can then be found using the xmi:id attribute. This contains a reference to the node, which can be used to extract the requirement text in a second step.

With the steps mentioned in **Error! Reference source not found.**, the extraction of the relevant information within the SysML diagram is completed. Hence, the transformation into a discrete event model needs to be conducted. Initially, it is essential to examine the structure of the simulation model file. The models are stored in XML format and can therefore be opened and analysed with any common XML viewer. The core elements for building an executable simulation model are all subordinated to the node <Model>. Individually for each model, the model's name is specified, the name of the Java package, and the model time. A node <DestroyCode> which contains executable Java source code as text content is executed as soon as an agent is destroyed e.g., when it reaches a Sink element, which defines the end of the process. This is required for the simulation process. Mandatory connectors must be defined in additional node. Data Set from the analysis library is transferred into a node to store recorded simulation data and give feedback to the user. Each of the encoded embedded elements is enclosed by a node. Furthermore, the name of the element appearing in the model layout within the simulation environment is specified by a unique name. Following that, the coordinates determining the position of an element need to be provided. This must be conducted in an additional node since the position is relevant to generate a human readable simulation interface. While it is not necessary to generate feedback in an 'autonomous' process, it is highly recommendable to define these locations. Furthermore, additional nodes, which are containing the element type

must be accessed. Lastly the behaviour of each element type can be adjusted by several parameters. These parameters are defined within the node. For example, with the aid of the parameter ‘onEnter’ it is possible to trigger certain behaviour when an agent enters the element. In this case, the current model time is saved to the Data Set using Java source code. Each of these elements is enclosed by connector nodes. The first one defines the name of the element. Upon that, two additional nodes must be created, to determine that element which is joined with the connector through its ‘out’ port. In contrast, target reference nodes determine the element which is joined with the connector through its ‘in’ port. Finally, as already mentioned the points nodes contains the coordinates of the corresponding ports.

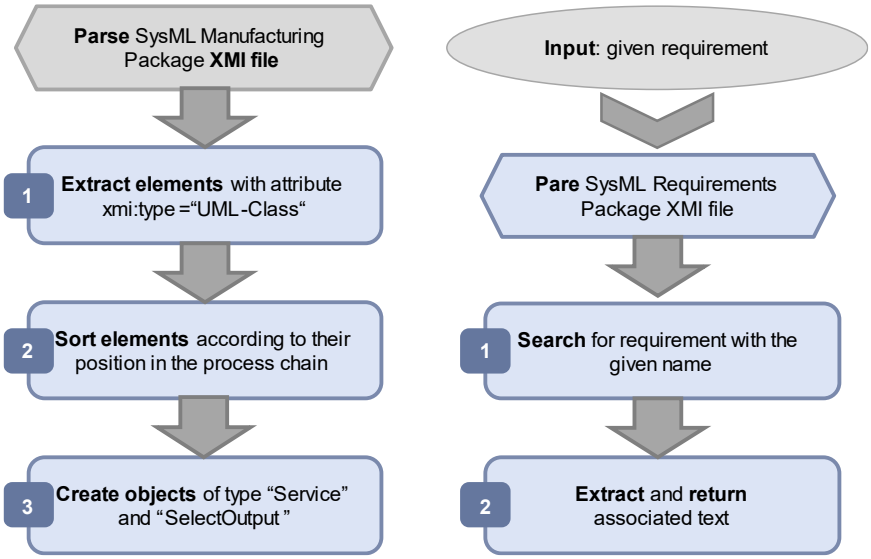

Figure 4: Extraction Approach for Manufacturing Structure and Assembly Requirements

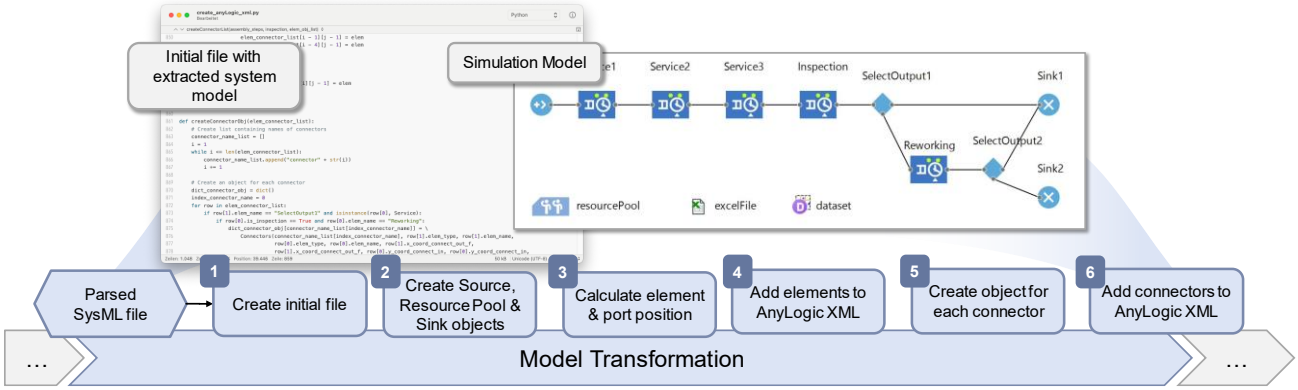

Figure 5: Script for the Model Transformation from SysML to Discrete Event Simulation

The process delivers a corresponding initial XML file (**Error! Reference source not found.**, step 1) compliant to the structure of an AnyLogic model and is executable for being simulated. The first step is to build the basic skeleton of the AnyLogic XML file. Except for the connector nodes and embedded objects, this initial file carries all the necessary data. The next step (2) is to create the Python objects to complete the simulation model in addition to those objects extracted from the SysML XMI files. At first, these are one object each of the type Source, Resource and Sink. This initiates, maintains and ends the simulation. If an inspection cycle is required, an additional object of the type ‘Sink’ must be generated. This can be easily checked on the associated Service objects since these are provided with a corresponding attribute when they are created. In the third step, the position of both the embedded objects and their ports is determined in the form of X and Y coordinates. The calculation of the element position is necessary to place the elements on the AnyLogic worksheet. The coordinates of the input and output ports are needed to finally connect the elements via connectors. The positioning of the elements is not crucial for the execution of the simulation but should still be chosen in reasonable ranges for improved visibility. A fixed relation between the position coordinates of each element type and its ports allows the derivation of the port coordinates based on the element position. In the next step (4), the AnyLogic XML file is extended with all the required embedded objects, such as Source, Service, SelectOutput, Sink and Resource Pool by adding the appropriate XML code. Afterwards, a Python object is created for each connector (5), specifying its source element, target element, and associated coordinates. In the last step (6), the code required to represent the connectors is

added to the AnyLogic XML file. This completes the creation of the XML model file as seen in **Error! Reference source not found.**. In total the simulation model comprises 11 embedded objects as well as both one Excel File and Data Set element. ‘Service1’, ‘Service2’ and ‘Service3’ represent the three required process stations with its specific delay times to assembly the angle grinder. The Service elements ‘Inspection’ and ‘Reworking’ together with ‘SelectOutput1’ and ‘SelectOutput2’ characterize the inspection cycle. ‘Sink1’ is used to record the point of time when a finished product exits, while ‘Sink2’ takes products which have been rejected.

3.1 Utilization Concept and Modelling Recommendations

The linking of the system model with assembly simulation is an example for model transformation within the concept of system model utilization. However, with this transformation the early system architecture can be verified towards specific assembly and production constraints. As seen, some of the information must be extracted from the domain-specific models, since the initial system model does not contain sufficient information for these simulations. However, this information from other domains can be integrated through bidirectional model synchronization as shown by Hick et al. (2023). With given information e.g., for new product generations that reuse or recycle existing system model parts, this information can directly be used for feedback on the system architecture as shown in Figure 6. The integration into the concept might enable the identification of synergies for other use cases and classes to increase the value of the created system models.

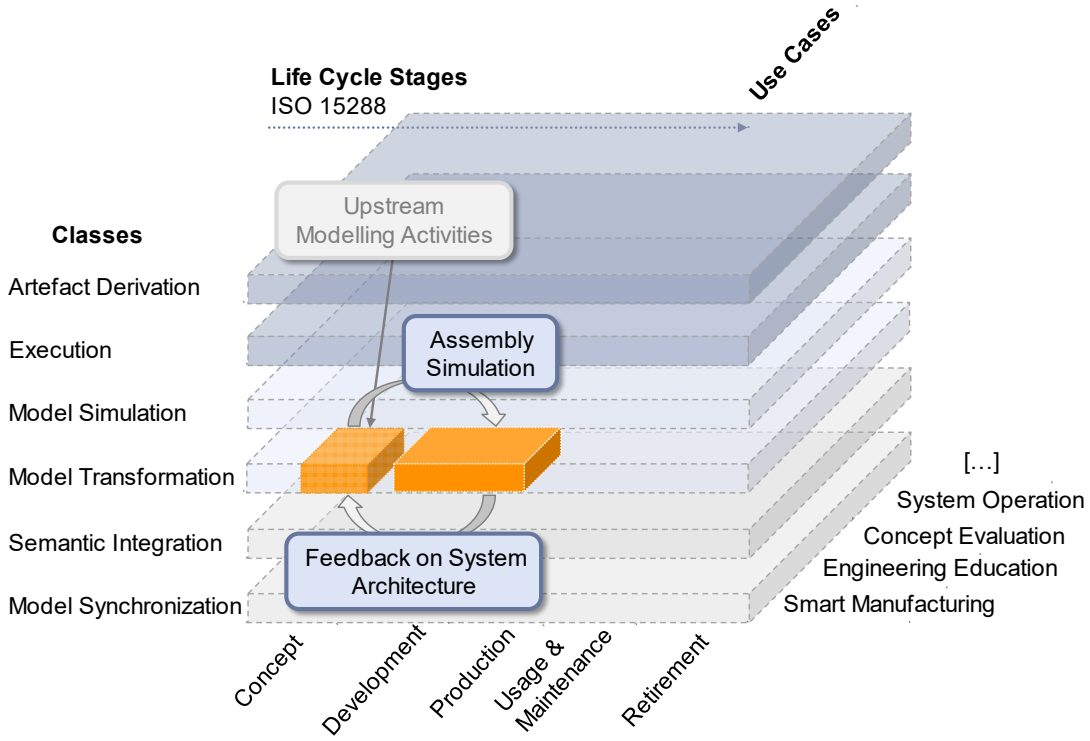

Figure 6: Classification of the System Model Utilization for the Digital Assembly Twin based on (Wilking et al., 2024)

To ensure that models are executable and prepared for utilization, specific modelling guidelines must be fulfilled. Thus, this can be the use of specific profiles within SysML to ensure a consistent approach. Nevertheless, it can be stated that at the beginning of requirements modelling, the focus should already be on production-specific characteristics. While these are not decisive for the creation of the Manufacturing Structure diagram, they are essential for the subsequent simulation process. In large and complex systems, it may be advisable to create a separate view exclusively for these types of requirements. In addition, it is important to ensure that requirements are named consistently and unique to be processed without complications by the transformation script.

When modelling the functional model, there are peculiarities in relation to the Manufacturing Structure diagram. Typically, in addition to an overview and hierarchy of functions, their detailed processes are also created in other views. However, it should be highlighted that in the context of the Manufacturing Structure diagram such a level of detail can be mostly disregarded. Of particular importance are the views for the overview and hierarchy of the system functions, since these indicate components that are important for the assembly process. Consequently, when modelling these views, emphasis must be placed on a structure that allows easy derivation of relevant blocks. The crucial factor in modelling the physical model is a structure that breaks down the physical architectural elements of the overall system under consideration into individual subsystems. Across several levels of abstraction, a model can thus be built that already considers the idea of the underlying manufacturing structure. It is important to emphasize that views should exist that make the structure of the system clearly comprehensible. Each of those views, for instance, can contain the structure of a component of the overall

system. However, this requires modelling that must go far into detail and requires significant domain knowledge which must be carried out by a system modeler who might not have this knowledge and must collaborate with an expert in this field. Only in this way it is possible to infer the structure of the assembly process of a system.

4 Concept for an Enlargement of the Digital Assembly Twin

The aforementioned concept enables the early verification through transferring relevant assembly information into a simulation. Moreover, this helps the systems engineer to avoid significant miscalculations during the set-up of the system architecture. But as Söderberg et al. (2017) state, a real time controller is highly relevant to grasp the full complexity of mechanisms during pre- and postproduction, such as tolerances, to ensure high product quality and full utilization of production capabilities. With the machine readability of the presented approach in an XMI structure, real time operational data can be linked with the simulation model and through the bidirectionality with the system model itself, as shown in Figure 7.

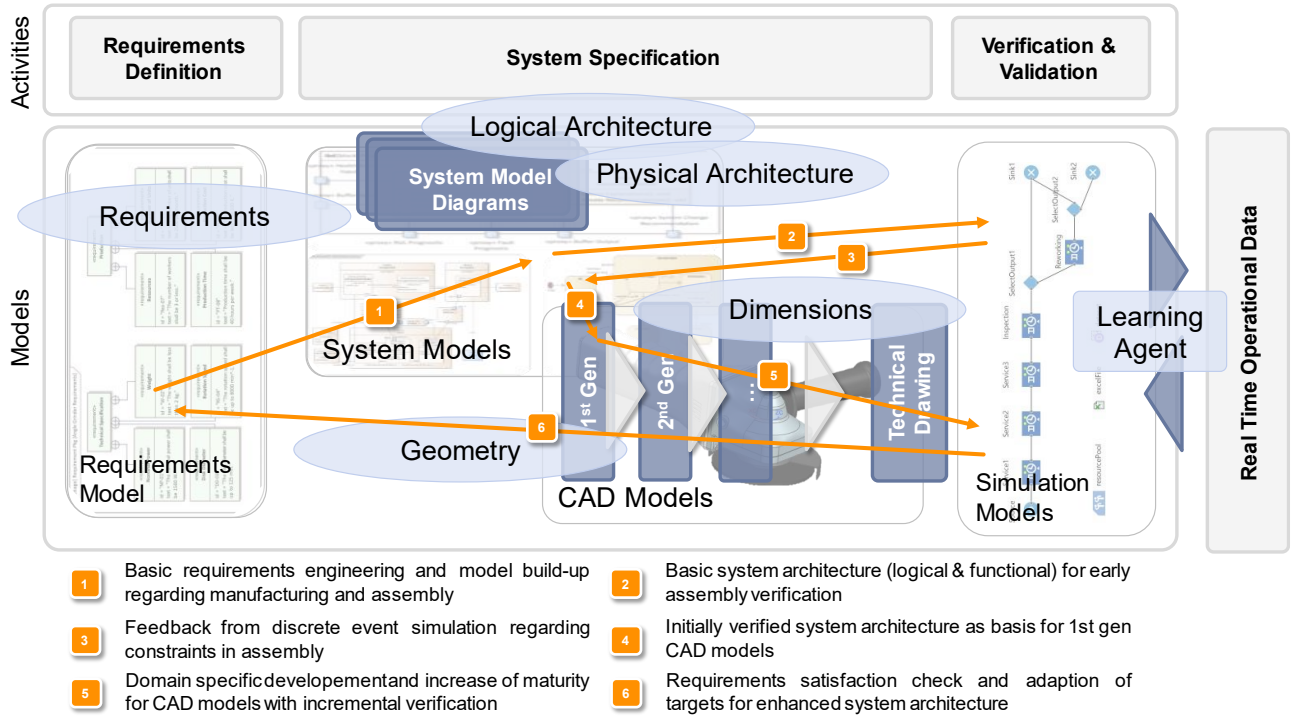

Figure 7: Models and Activities for the Digital Assembly Twin based on (Hick et al., 2023)

While initial requirements were used to build up the system architecture, early simulations regarding the assembly and its feedback, might influence the architecture and lead to a change of it. With this feedback the system architecture is used as a basis for a first generation of CAD models. When the system and its components are in production and assembly, real time operational data can be used through a learning agent that connects the actual assembly with the simulation. Through the bidirectional connection of the assembly simulation model, a link and requirements satisfaction check can be conducted. This allows the systems engineer to monitor whether the initial requirements regarding the production phase are fulfilled. Deviations from this can directly be linked to relevant system model elements within the manufacturing structure to support the decision-making process, whether the system architecture needs to be changed or whether individual components are affected. This involves the whole system architecture into the feedback loop, allowing a structured top-down iteration to prevent an unwanted impact propagation of changes for individual components that likely result from a bottom-up approach.

5 Conclusion and Outlook

The presented approach shows an integration of a manufacturing structure into the system architecture. It is crucial to harmonize this concept with existing development approaches and reduce the additional modelling effort coming along the presented approach. With the utilization of SysML system models, it was shown that such an approach is viable, answering the first research question. However, the approach was conducted for a simple product. Larger and more complex systems will require further integration of the approach and a further evaluation of the used profile and model transformation mechanism. This does not seem impossible but must be evaluated further regarding its feasibility. Nevertheless, complex systems regarding their assembly can be segmented in smaller, less complex subsystems. With

this, the approach can be limited to individual subsystems to achieve a quick benefit for utilizing system models. Especially the reuse of components with known manufacturing structures will help to quickly build up an understanding of implications regarding the production process and will reduce the risk of miscalculations for the systems engineer in early stages of the system architecture development. With the introduction of SysML v2 a new API will reduce the effort coming along with a partial XMI extraction parser, making this approach more feasible.

Furthermore, enlarging this concept towards a Digital Twin seems reasonable, as no additional modelling language is introduced. This allows to keep consistency throughout the modelling language for the system architecture. However, this comes along with the limitations of the language and other languages, which are more focused on the description and implementation of Digital Twins (Corradini et al., 2023). This could support the introduction of software platforms for Digital Twins that combine different aspects, such as IoT and models (Tao et al., 2024). Still, a concept further involving a Digital Twin into production and assembly activities to improve the quality of the system is reasonable but must be pondered in consideration of high costs or these automated assembly lines.

Acknowledgement

This Project is supported by the Federal Ministry for Economic Affairs and Climate Action (BMWK) on the basis of a decision by the German Bundestag. It is part of the IGF Project 22467 BG (FVA 889 II Digital Twin) in collaboration with the Forschungsvereinigung Antriebstechnik (FVA) e.V.

Supported by:

Federal Ministry
for Economic Affairs
and Climate Action

on the basis of a decision
by the German Bundestag

References

Albers, A., Bursac, N., Wintergerst, E., 2015. Product Generation Development - Importance and Challenges from a Design Research Perspective, in: Mastorakis, N.E., Solomon To, C.W. (Eds.), Proceedings of the International Conference on Mechanical Engineering (ME 2015), Vienna, Austria, March 15-17, 2015, pp. 16–21.

Brahmi, R., Hammadi, M., Aifaoui, N., Choley, J.-Y., 2021. Interoperability of CAD models and SysML specifications for the automated checking of design requirements. Procedia CIRP 100, 259–264. https://doi.org/10.1016/j.procir.2021.05.064.

Campo, K.X., Teper, T., Eaton, C.E., Shipman, A.M., Bhatia, G., Mesmer, B., 2023. Model-based systems engineering: Evaluating perceived value, metrics, and evidence through literature. Syst Eng 26, 104–129. https://doi.org/10.1002/sys.21644.

Ciccozzi, F., Malavolta, I., Selic, B., 2019. Execution of UML models: a systematic review of research and practice. Software & Systems Modeling 18, 2313–2360. https://doi.org/10.1007/s10270-018-0675-4.

Corradini, F., Fedeli, A., Fornari, F., Polini, A., Re, B., 2023. DTMN a Modelling Notation for Digital Twins, in: Sales, T.P., Proper, H.A., Guizzardi, G., Montali, M., Maggi, F.M., Fonseca, C.M. (Eds.), Enterprise Design, Operations, and Computing. EDOC 2022 Workshops, vol. 466. Springer International Publishing, Cham, pp. 63–78.

Czwick, C., Martin, G., Anderl, R., Kirchner, E., 2020. Cyber-Physische Zwillinge. ZWF 115, 90–93. https://doi.org/10.3139/104.112310.

Fett, M., Wilking, F., Goetz, S., Kirchner, E., Wartzack, S., 2023. A Literature Review on the Development and Creation of Digital Twins, Cyber-Physical Systems, and Product-Service Systems. Sensors (Basel, Switzerland) 23. https://doi.org/10.3390/s23249786.

Friedenthal, S., Dori, D., Mordecai, Y., 2021. SEBoK - Why model? https://sebokwiki.org/w/index.php?title=Why_Model%3F&oldid=67848 (accessed 18 October 2023).

Giese, H., Hildebrandt, S., Neumann-Paulick, S., 2009. Towards Integrating SysML and AUTOSAR Modeling via Bidirectional Model Synchronization, in: Dagstuhl-Workshop MBEES: Model-based Development of Embedded Systems, Braunschweig. 22.-24.04.2009, pp. 155–164.

Girard, G., Baeriswyl, I., Hendriks, J.J., Scherwey, R., Müller, C., Hönig, P., Lunde, R., 2020. Model based Safety Analysis using SysML with Automatic Generation of FTA and FMEA Artifacts, in: Proceedings of the 30th European Safety and Reliability Conference and 15th Probabilistic Safety Assessment and Management Conference. Proceedings of the 29th European Safety and Reliability Conference (ESREL). 01.11.2020 - 05.11.2020. Research Publishing Services, Singapore, pp. 5051–5058.

Henderson, K., Salado, A., 2021. Value and benefits of model-based systems engineering (MBSE): Evidence from the literature. Syst Eng 24, 51–66. https://doi.org/10.1002/sys.21566.

Hick, H., Bajzek, M., Faustmann, C., 2019. Definition of a system model for model-based development. Applied Sciences 1. https://doi.org/10.1007/s42452-019-1069-0.
Hick, H., Sanladerer, S., Trautner, J., Ryan, K., Piguet, J., Wilking, F., Horber, D., Faustmann, C., Kranabitl, P., Kollegger, S., Bajzek, M., Schleich, B., Wartzack, S., 2023. Combining System Models and CAD for Change Scenario Management. INCOSE International Symp 33, 117–132. https://doi.org/10.1002/iis2.13012.
Kruse, B., Blackburn, M., 2019. Collaborating with OpenMBEE as an Authoritative Source of Truth Environment. Procedia Computer Science 153, 277–284. https://doi.org/10.1016/j.procs.2019.05.080.
Liu, Y., Irudayaraj, P., Zhou, F., Jiao, R.J., Goodman, J.N., 2014. SysML-based Model Driven Discrete-Event Simulation, in: Moving Integrated Product Development to Service Clouds in the Global Economy. IOS Press, pp. 617–626.
Lu, Y., Liu, C., Wang, K.I.-K., Huang, H., Xu, X., 2020. Digital Twin-driven smart manufacturing: Connotation, reference model, applications and research issues. Robotics and Computer-Integrated Manufacturing 61, 101837. https://doi.org/10.1016/j.rcim.2019.101837.
Malik, A.A., Bilberg, A., 2018. Digital twins of human robot collaboration in a production setting. Procedia Manufacturing 17, 278–285. https://doi.org/10.1016/j.promfg.2018.10.047.
Morkevicius, A., Jankevicius, N., 2015. An approach: SysML-based automated requirements verification, in: 2015 IEEE International Symposium on Systems Engineering (ISSE). 2015 IEEE International Symposium on Systems Engineering (ISSE), Rome, Italy. 28.09.2015 - 30.09.2015. IEEE, pp. 92–97.
Palachi, E., Cohen, C., Takashi, S., 2013. Simulation of cyber physical models using SysML and numerical solvers, in: 2013 IEEE International Systems Conference (SysCon). 2013 7th Annual IEEE Systems Conference (SysCon), Orlando, FL. 15.04.2013 - 18.04.2013. IEEE, pp. 671–675.
Qi, Q., Tao, F., 2018. Digital Twin and Big Data Towards Smart Manufacturing and Industry 4.0: 360 Degree Comparison. IEEE Access 6, 3585–3593. https://doi.org/10.1109/ACCESS.2018.2793265.
Rezaei Aderiani, A., Wärmefjord, K., Söderberg, R., 2021. Evaluating different strategies to achieve the highest geometric quality in self-adjusting smart assembly lines. Robotics and Computer-Integrated Manufacturing 71, 102164. https://doi.org/10.1016/j.rcim.2021.102164.
Schleich, B., Anwer, N., Mathieu, L., Wartzack, S., 2017. Shaping the digital twin for design and production engineering. CIRP Annals 66, 141–144. https://doi.org/10.1016/j.cirp.2017.04.040.
Schleich, B., Wärmefjord, K., Söderberg, R., Wartzack, S., 2018. Geometrical Variations Management 4.0: towards next Generation Geometry Assurance. Procedia CIRP 75, 3–10. https://doi.org/10.1016/j.procir.2018.04.078.
Schoenherr, O., Pappert, F.S., Rose, O., 2014. Domain Specific Simulation Modeling with SysML and Model-to-Model Transformation for Discrete Processes, in: Fonseca i Casas, P. (Ed.), Formal Languages for Computer Simulation. IGI Global, pp. 267–304.
Schonherr, O., Rose, O., 2009. First steps towards a general SysML model for discrete processes in production systems, in: Winter Simulation Conference. 2009 Winter Simulation Conference (WSC 2009), Austin, TX, USA. 12/13/2009 - 12/16/2009. ACM Digital Library, pp. 1711–1718.
Schumacher, T., Inkermann, D., 2022. Heterogeneous models to Support Interdisciplinary Engineering - Mapping Model Elements of SysML and CAD. Procedia CIRP 109, 653–658. https://doi.org/10.1016/j.procir.2022.05.309.
Söderberg, R., Wärmefjord, K., Carlson, J.S., Lindkvist, L., 2017. Toward a Digital Twin for real-time geometry assurance in individualized production. CIRP Annals 66, 137–140. https://doi.org/10.1016/j.cirp.2017.04.038.
Stark, R., Anderl, R., Thoben, K.-D., Wartzack, S., 2020. WiGeP-Positionspapier: „Digitaler Zwilling“. ZWF 115, 47–50. https://doi.org/10.3139/104.112311.
Tao, F., SUN, X., CHENG, J., ZHU, Y., LIU, W., WANG, Y., XU, H., HU, T., LIU, X., LIU, T., SUN, Z., XU, J., BAO, J., XIANG, F., JIN, X., 2024. makeTwin: A reference architecture for digital twin software platform. Chinese Journal of Aeronautics 37, 1–18. https://doi.org/10.1016/j.cja.2023.05.002.
Wagner, H., Portenlänger, L., Zuccaro, C., 2023. Using SysML Models as Digital Twins for Early Validation of Modular Systems and Systems of Systems, in: 2023 18th Annual System of Systems Engineering Conference (SoSe). 2023 18th Annual System of Systems Engineering Conference (SoSe), Lille, France. 6/14/2023 - 6/16/2023. IEEE, pp. 1–7.
Wärmefjord, K., Söderberg, R., Schleich, B., Wang, H., 2020. Digital Twin for Variation Management: A General Framework and Identification of Industrial Challenges Related to the Implementation. Applied Sciences 10, 3342. https://doi.org/10.3390/app10103342.
Wilking, F., Horber, D., Goetz, S., Wartzack, S., 2024. Utilization of System Models in Model-Based Systems Engineering - Definition, Classes and Research Directions based on a Systematic Literature Review. Design Science (Accepted).
Wilking, F., Sauer, C., Schleich, B., Wartzack, S., 2022. SysML 4 Digital Twins – Utilization of System Models for the Design and Operation of Digital Twins. Proc. Des. Soc. 2, 1815–1824. https://doi.org/10.1017/pds.2022.184.
Wu, S.-Y.D., Wysk, R.A., 1989. An application of discrete-event simulation to on-line control and scheduling in flexible manufacturing. International Journal of Production Research 27, 1603–1623. https://doi.org/10.1080/00207548908942642.
Zheng, Y., Yang, S., Cheng, H., 2019. An application framework of digital twin and its case study. J Ambient Intell Human Comput 10, 1141–1153. https://doi.org/10.1007/s12652-018-0911-3.

Contact: Fabian Wilking, Friedrich-Alexander-Universität Erlangen-Nürnberg, Engineering Design, Martensstrasse 9, 91058 Erlangen, Germany, wilking@mfk.fau.de