Model-Driven Systems Engineering for Virtual Product Design

Manuela Dalibor, Nico Jansen, Bernhard Rumpe, Louis Wachtmeister, Andreas Wortmann
Software Engineering, RWTH Aachen University, http://www.se-rwth.de/

***Abstract*—Model-Based Systems Engineering (MBSE) aims to provide a systems engineering methodology that leverages modeling methods to support design, analysis, verification, and validation of systems. As such, methodologies for MBSE have to be able to integrate heterogeneous engineering models from a variety of domains, including mechanical engineering, product design, legislation, and many more. Most research in this area usually only focuses on general system descriptions of a step in the system development process, without providing any interdisciplinary or interprocess connections. Thus, the models created by the domain experts are often unconnected and not suited for automated model transformations. We present a method to integrate abstract system descriptions in the Systems Modeling Language (SysML) with Computer-Aided Design (CAD) models, which are not only the primary model for geometrical hardware descriptions but also the starting point of most process chains in the context of virtual product development. The transformations of our method are realized in a plug-in for the MagicDraw modeling environment and support to generate parameters of a parameter and constraint CAD modeling approach from a SysML model. This automates the integration of abstract system descriptions with design models to foster a coherent virtual product development.**

***Index Terms*—Model-Driven Systems Engineering, SysML, CAD, Tool Integration**

I. Introduction

The engineering of cyber-physical systems demands the interdisciplinary collaboration of experts from a variety of different domains, including mechanical engineering, product design, legislation, *etc.* to produce software artifacts. However, these are rarely software engineering experts and, hence, need to overcome the conceptual gap [1] between their domain of expertise and software development. Modeling languages [2] can reduce this gap by using syntax and semantics closer to the domain of interest than to computer programming. Consequently, research and industry are increasingly turning to Model-Based Systems Engineering (MBSE) to leverage models for the description of systems, communication, and discussion of designs, and exchange between experts. While yielding advantages over "traditional" document-based systems engineering, praxis has shown the established use of models in MBSE usually is too vague to leverage their potential for automation [3].

In systems engineering, software, and hardware components are often produced in parallel. Errors detected during their integration, hence, are more costly than errors detected earlier. To overcome this problem, virtual product development describes a practice to support all phases of the product development process using a digital environment [4]. For this aim, it applies computer-aided modeling methods in all development-relevant phases of the product life-cycle to simulate, verify, validate, and manufacture the product while minimizing the creation of physical prototypes. The virtual development process comprises three activities: (1) Development of a virtual product design that describes the geometry of hardware elements using 2D or 3D geometric modeling environments *e.g.,* by creating CAD models in a CAD suite. (2) Virtual product simulation, which analyses the product in a simulation environment; and (3) Virtual manufacturing creates the actual hardware product.

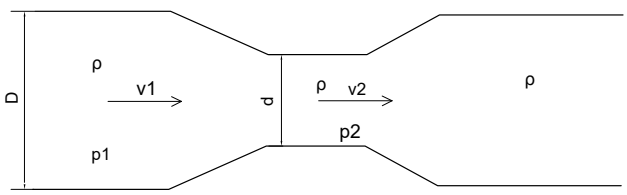


Fig. 1: A conceptual sketch of a Venturi nozzle showing the most important flow parameters for the velocity, pressure, and density

While the connection between the virtual product design, analysis, and digital manufacturing is a research topic in classical mechanical engineering [4], [5], the integration between abstract artifacts created in MBSE often is underexplored there. Therefore, we present a small Model-Driven Systems Engineering (MDSE) methodology that aims to integrate the functional SysML paradigm with the geometric CAD paradigm. Our contribution, therefore, is

- a process to integrate SysML models in a virtual product development environment,
- modeling methods to support this process,
- and a tool to implement the SysML-CAD connection of this process.

In the following, Sec. II introduces a typical example from mechanical engineering. Afterwards, Sec. III introduces MDSE and virtual product development. Then, Sec. IV introduces a method for virtual product design that uses process chains to integrate SysML models with CAD models. Sec. V describes how to apply newly developed and existing tools

This research has partly received funding from the German Research Foundation (DFG) under grant no. EXC 2023 / 390621612. The responsibility for the content of this publication is with the authors.

to our application example to create a virtual product design. Finally, Sec. VI discusses observations, Sec. VII highlights related work, and Sec. VIII concludes.

II. Example

Consider the development of a Venturi nozzle that serves as vacuum creator in the pneumatic system of an industrial robot. This robot uses its pneumatic system to operate suction cups for pick and place operations.

Fig. 1 shows a concept drawing and the primary model parameters that are required to perform first mathematical analyses and describe the essential physical principles. Nozzles of this kind can serve as vacuum creators since the Venturi effect causes that the static pressure p_1 at the inlet section of the nozzle with diameter D decreases to p_2 when the tube diameter is decreased to d. At the same time, the flow speed v_1 increases to v_2. To simplify the simulations and the definition we further assume that the flow is incompressible, which means that the density ρ stays constant in all parts of the nozzle.

Even though this simple mechanical component has no complex interactions with software or electrical parts, the models to describe, simulate, and manufacture this component are surprisingly diverse and complicated. The main challenge of this example is that these various models are usually created by different engineers of heterogeneous domains, using multiple modeling tools and techniques. For systems, such as the pick-and-place robot, often SysML models are used to describe an abstract view of this system.

Based on this description, CAD models have to be created to describe the hardware design in a graphical 2D or 3D environment. Hence, in addition to the heterogeneity of the models, also the model creation tools that are used in different engineering domains are heterogeneous.

Thus, many engineers of different domains use various modeling tools to create multiple model-views on different abstraction levels. While mechanical engineers mostly use CAD modeling tools such as PTC Creo, CATIA, or Autodesk Inventor, visual modeling tools for software and systems such as PTC Modeler, Enterprise Architect, or NoMagic's MagicDraw are often used by software and systems engineers to describe abstract system designs. For that reason, it becomes necessary to exchange parameters between these different modeling tools to distribute (and validate) the information among various domain experts.

We present a method to exchange important model parameters such as dimensions between a MagicDraw SysML model and Autodesk Inventor CAD model. Additionally, we use this CAD model to simulate the behavior of the system and produce a hardware prototype.

III. Preliminaries

Our method for MDSE aims to facilitate consistency checking between functional SysML models and geometric CAD models by enabling bidirectional transformations between both

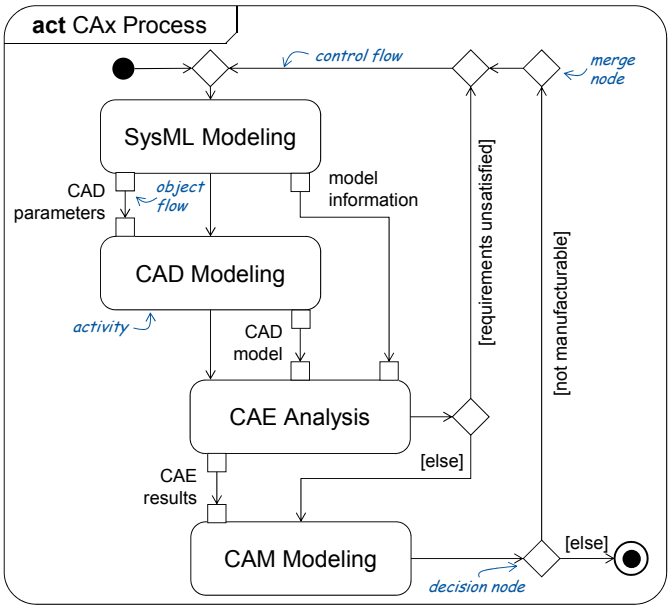


Fig. 2: An iterative system development process that uses SysML as starting point to enable CAx process activities

paradigms. There are various definitions for systems engineering [6]–[9]. Some emphasize its focus on a whole system in contrast to its parts [6], whereas others describe it as an iterative top-down process that aims at developing a system that fulfills all requirements [7], or as an interdisciplinary approach for developing successful systems [8]. Model-Based Systems Engineering (MBSE) is a novel paradigm of systems engineering that applies formalized modeling principles, methods, languages, and tools to the entire life-cycle of complex systems [10]. In contrast to traditional document-based engineering, in MBSE models are the primary development artifacts. Such models can use domain terminology the experts are familiar with. They are more abstract than documents to facilitate reuse, and more formal to enable automated analysis and consistency checking. Model-Driven Systems Engineering (MDSE) even extends the idea of MBSE and aims at automating parts of the development process leveraging models. To achieve this aim, models of a MDSE methodology must not only be the key artifacts of the engineering process but also sufficient to generate (*e.g.,* by model refinement or transformation) other descriptions that address system requirements, design, analysis, verification, and validation activities.

The OMG Systems Modeling Language (SysML) [11] has become a de-facto standard for specifying system parts in the development process [12], [13]. SysML is a graphical modeling language that enables engineers to model the system architecture and facilitates the application of MBSE, *e.g.,* by supporting engineers in defining the system structure, dependencies between system parts, the system behavior, and connecting these with requirements.

SysML is a subset of the Unified Modeling Language (UML) [14] extended with functionality for systems engineering that is realized as a graphical modeling language. It enables

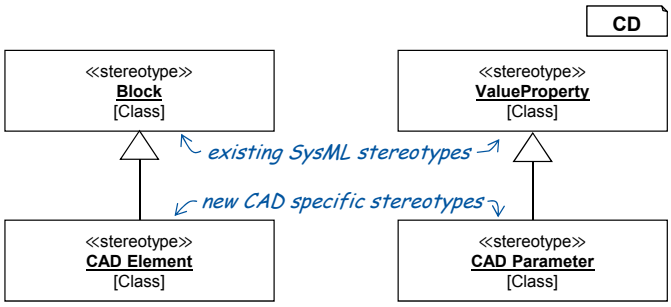


Fig. 3: The CAD profile extension that includes «CAD Element» and «CAD Parameter» into SysML

engineers to model the system architecture and behavior and facilitates the application of MBSE, *e.g.,* by supporting the integration of structure with behavior and constraints. Hence, the SysML features four diagram types: (1) Structure: Block Definition Diagrams (BDDs), Internal Block Diagrams (IBDs), and Package Diagrams; (2) Behaviour: SysML provides Sequence Diagrams, State Machines, and Activity Diagrams; (3) Parametrics (constraints): parametric diagrams to specify physical properties and dependencies between parameters; and (4) Requirements: diagrams that provide a graphical notation to capture requirements of a system.

The Computer-Aided x (CAx) paradigm describes various computer-aided methods, which enable system engineers to define concrete products in their respective engineering domain. Typical CAx models are for example, CAD, Computer-Aided Engineering (CAE), and Computer-Aided Manufacturing (CAM) models. Because CAD models describe the geometry and design of a product, they are the main artifact of a virtual product design process. Based on the CAD model, the CAE models analyze the product design concerning different engineering analysis criteria by using simulation methods such as Finite Element Method, Computional Fluid Dynamics (CFD), or Multibody Dynamics. Because simulations are the primary methods that engineers use in this context, the CAE analysis is strongly connected with the virtual product simulation. Finally, the CAM summarizes all computer-aided methods that are usable to manufacture the product and is related with digital manufacturing activities.

The combination of CAD, CAE, and CAM in the context of virtual product development leads to the so-called CAx process chains. A CAx process chain is a modeling or programming method to build a digital representation, to calculate visualizations, analyses, simulations, optimizations, or control data [5]. In these process chains, the CAD model has an important role, since it provides the digital hardware descriptions, which serve as the base of the data transformations that the CAx process chain performs.

IV. Modeling Process and Methods

In order to combine the advantages of a MDSE approach with the benefits of virtual product development, this section aims to introduce a development process based on process chains and describes methods to implement this process.

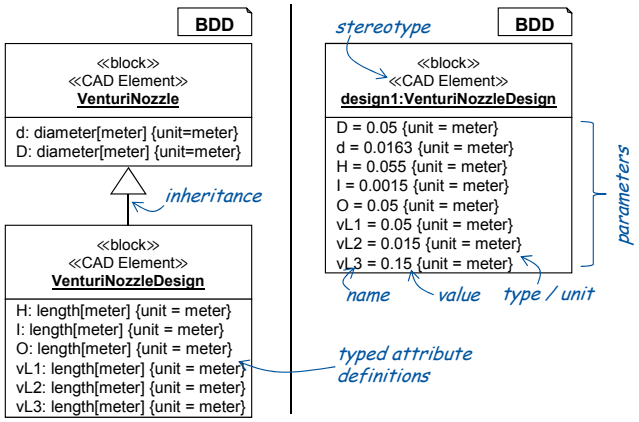


Fig. 4: The general Venturi nozzle BDD with parameters specifying its geometric dimensions (left) and concrete Venturi nozzle instance (right)

Therefore, we connect the idea of CAx process chains with SysML models created using the Object-Oriented Systems Engineering Method (OOSEM) [15].

The resulting process that Fig. 2 presents as SysML Activity Diagram, renders a coherent virtual product development approach. Using this process systems engineers are enabled to describe the system in SysML and integrate this model into CAx process chains for virtual product development. For this, the process starts with a SysML modeling activity, in which the systems engineer creates an abstract system model by using BDDs to specify elements of the physical parts of the system. Next, the process forwards parameters of the BDDs that are relevant for the CAD modeling activity. By using the resulting CAD model in combination with additional information from the SysML diagram, the process enables the engineers to perform additional CAE analyses and CAM modeling in the respective activities based on CAD-CAE or CAD-CAM process chains. Typical additional information is, for instance, dimension parameters for geometric models, environment information for simulation, or materials and tolerance information for the production. Since the first drafts of the product might not meet the requirements or are not manufacturable, the activity diagram from Fig. 2 has decision nodes which lead back to the SysML Modeling activity to implement an evolutionary development.

Since the connection between CAD, CAE, and CAM is already covered in the context of CAx process chains [4], [5], we will focus on the implementation of this SysML-CAD connection in the following.

As the process depicted in Fig. 2 forwards the relevant parameters from SysML to CAD, it is necessary to identify a construction theoretical foundation for the creation of the CAD model. Since parameter and constraint modeling techniques enable engineers to store the dimension parameters independently from the geometrical product description models [5], this construction theoretical approach seems to be suited for connection dimension parameters from a SysML model with a graphical representation. By this, it is possible to separate

Name	Causality	Value	Type
Venturi Nozzle Flow Analysis			
A_1	target	0.002	m^2
A_2	target	$2.087 \cdot 10^{-4}$	m^2
C	given	0.98	$-$
D	given	0.05	m
d	given	0.0163	m
p_1	given	$594,037.000$	Pa
p_2	target	$12,016.428$	Pa
ρ	given	1.225	kg/m^3
v_1	given	101.619	m/s
v_2	target	980.084	m/s

TABLE I: Flow analysis results of the concrete Venturi nozzle design from Fig. 4

parameter names, their value or expressions describing their value, and their unit from the actual geometric model. Additionally, we have to keep in mind that it might be unrealistic to assume that the dimension parameters for the CAD model and the object geometry are modeled entirely independent from each other. Thus, it might be helpful to include a reverse direction in this process, which allows returning parameters from a CAD model to SysML again.

Because not all instances of system blocks must represent hardware elements with a corresponding CAD model, we require a method to mark the blocks and their value properties of a BDD as parameters. To implement this, we can use specific profiles that introduce new stereotypes for SysML elements. Fig. 3 presents a SysML profile extension, which enables systems engineers to mark blocks as «CAD Element» to indicate that this block also requires the creation of a CAD model. Besides, not all value properties of the block are necessarily relevant parameters for the CAD model as well, hence the «CAD Parameters» stereotype for value properties indicates that this property is a CAD parameter.

V. Case Study and Tool Presentation

To visualize and apply the methods we developed in the previous chapter, we modeled the Venturi nozzle as described in Sec. II. Additionally, we describe how newly developed and already existing tools support the CAx process we presented in Fig. 2. We began the modeling process of the Venturi nozzle example by creating a block definition diagram for a general nozzle and extended this abstract description in a second step to a concrete Venturi nozzle design using MagicDraw as shown in Fig. 4. To derive a concrete instance of the Venturi nozzle, we modeled the physical flow in the nozzle in a parametric diagram. This parametric diagram enables us to express parameter relationships and descriptions we require to run first simulations to estimate dimension parameters that create a sufficient vacuum to operate a specific suction cup. Based on the relationships described in the parametric diagram, we then used MagicDraw's ParaMagic plug-in and the OpenModelica solver to determine and test multiple designs until we found a suited Venturi nozzle design. The results of this computation for the final design are given in Fig. 4.

In parallel, we developed a MagicDraw plug-in which exports instances of «CAD Element» blocks into a parameter

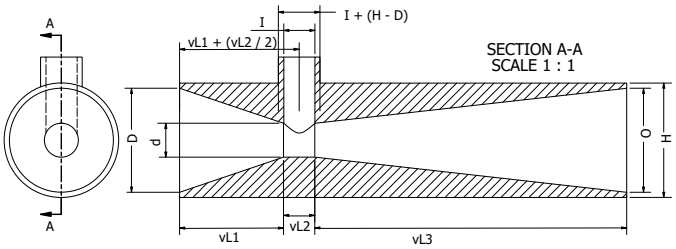


Fig. 5: Technical drawing of a Venturi nozzle with parameters as dimension placeholders

description file and imports updates of this file again into the MagicDraw block instances. The plug-in is structured according to the component diagram Fig. 7 presents. This CAD plug-in for MagicDraw subdivides into a `MDHandler` component that processes the model tree MagicDraw provides, a `SIUnits` component that converts units from and to representations of other tools, and a `ParameterExport` component that is responsible for creating the actual parameter file. On the other side, parametric CAD suites such as Autodesk Inventor provide methods to handle and import parameters, create geometries that use these parameters, and to create technical drawings as standardized model representations.

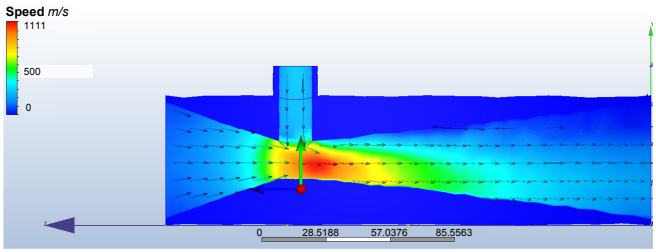
To process the Venturi nozzle instance Fig. 4 presents, the MagicDraw plug-in searches all CAD elements in the project and generates for each instance a parameter exchange file, which contains the parameter name, its value, and its unit. By connecting this file with a part drawing in Autodesk Inventor, we then modeled the geometry of the Venturi nozzle as shown in Fig. 5 and used the exchange file to define and compute the concrete instance of the Venturi nozzle based on the parameters.

To further verify the results of the Venturi nozzle, we then used Autodesk CFD to simulate the geometric model of the Venturi nozzle as a part of the CAE process. The results of this simulation splits into the flow velocity that Fig. 6a presents and the pressure, which Fig. 6b visualises. If we compare these results of the CFD analysis with our initial analysis based on the SysML model, we can see that the results are principally in the same range, but provide more detailed information about the velocity and pressure distribution in the nozzle design.

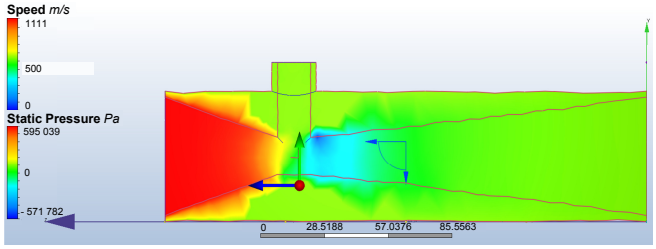
Finally, we used a 3D printer to print a hardware prototype of the Venturi nozzle, which we only used to check the functionality of the result, as the print quality and the lack of suited measurement technology did not allow any further validations about the simulation or the manufacturability of this hardware model.

VI. Discussion

The presented approach addresses several problems from multi-paradigm modeling, such as the handling of heterogeneous models and modeling languages by introducing a process chain that connects heterogeneous system models. An advantage of our integration is that it supports an evolutionary system development without breaking the traditional process



(a) Autodesk CFD Analysis Results (Velocity)



(b) Autodesk CFD Analysis Results (Pressure)

Fig. 6: Autodesk CFD Analysis Results of the Venturi Nozzle Design based on the Parameters specified in the instance from Fig. 4

of virtual product development design, simulation, and manufacturing. An evolutionary approach enables the continuous improvement of the system design over multiple iterations. Thus, the development information is not only forwarded to the next process activity but may also flow back from each development step to the previous process activity. By this, the engineer can iteratively improve the design based on flaws that the next process activity uncovers and even perform subsequent development steps in parallel. Since the connection between the SysML and the CAD model enables reimporting changed parameters from the CAD model into SysML again, the created models are reusable for the next development iteration. If the engineer that creates the SysML model forgets to define a parameter the engineer for the CAD model requires, the later can specify this parameter himself and use the CAD plug-in to reintegrate this parameter to the SysML model. Hence, it is possible to synchronize the SysML and the CAD model with reduced overhead compared to a manual update process. Moreover, this approach could be extended with additional automated checks for model consistency to decide whether the change the CAD engineer made are valid.

One issue we did not solve yet is that no data is returned from the CAE and CAM to the SysML model, to improve the quality of the next development iteration. Additionally, the usability of our process would improve even more, if requirements and general model sanity checks for the SysML-CAD process chain would be performed *e.g.,* by automatically checking whether all dimensions are within certain bounds and fit in their assembly based on given requirements.

In order to improve our approach, future works could investigate the mathematical connection between the different abstraction levels further, for instance by taking additional works about the theory behind systems engineering [16], principle solutions for the design of technical products [17], multi-view modeling and graph transformations for MBSE [18], [19], or system and simulation theories for multi-paradigm modeling [20] into account.

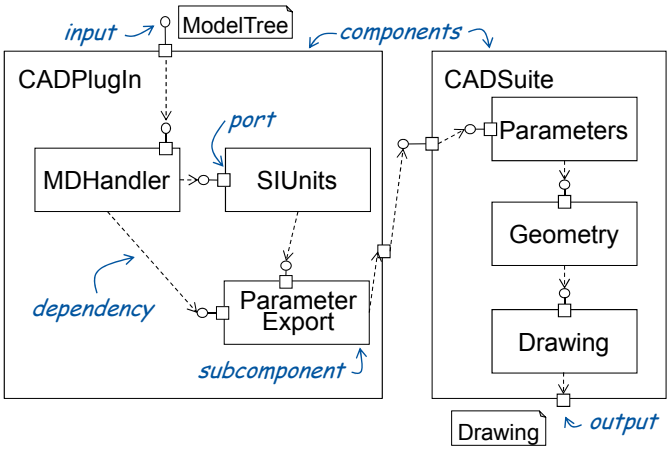


Fig. 7: Component Diagram of the plug-in structure

Finally, digital mock-ups could serve as virtual prototypes that are usable in all stages of the virtual product development approach [4]. An extension of the process developed in this work could be to investigate how this integration could make use of digital shadows and twins. By developing an integrated toolchain and connecting knowledge, we ultimately take steps towards a digital twin that could adapt automatically depending on the system's state and suggest design improvements.

VII. Related Work

Model integration in the MBSE process is subject to ongoing research. Our work relates to multiple research domains, such as systems engineering, virtual product design, and multi-paradigm modeling.

The approach presented in [21] employs SysML to organize and integrate heterogeneous models that contribute to a common system model. Relationships and dependencies between these models are formulated in SysML to achieve a consistent model. Similar to our solution, this approach involves constructing a high-level system model in SysML for integrating CAE methods. Additionally, they address the connection of SysML with simple engineering constraints and requirements for a CAE analysis. However, they do not address the creation of concrete CAD models, nor do they provide any tools to automate the integration.

Another methodology is more focused on virtual product development [4] and provides an extensive overview of model-based virtual product development methods. It is not only restricted to the modeling aspects of mechanical engineering but also introduces methods for the development of electrical or software systems. The processes cover a considerable number of modeling techniques for virtual product development and systems engineering. However, they do not render a general methodology to systematically integrate models, described in a systems modeling language such as SysML, with other artifacts of the virtual product development process.

In addition to the generative aspects, the approach in [22] explains how MBSE provides models to handle design information created and processed by multiple stakeholders of the systems engineering process. To achieve this goal, they use multiple tool plug-ins to generate and exchange model information between different tools automatically. Although the approach presents some general aspects, the developed tools mainly focus on the usage at Johnson Space Center and are not a general methodology.

Another elaboration provides a general overview of multi-paradigm modeling in combination with simulation [20]. Especially the concept of multi-formalism modeling is described, where interacting constituents are modeled using different formalism in a coupled model. Multi-formalism modeling is applied, when it is convenient or necessary to address several concerns within particular formalisms and to bridge different abstraction levels. Our method to integrate SysML diagrams and CAD models of the CAx process corresponds to this approach. SysML and CAD correspond to different formalisms, which together contribute to a target component. Connecting SysML and CAx thus enables the development of integrated heterogeneous models.

VIII. Conclusion

We presented a MDSE methodology for virtual product design and demonstrated its feasibility for virtual product development by modeling a small case study from mechanical engineering.

Our methodology consists of a general process for virtual product development, of which the SysML-CAD process chain has been implemented to support the creation of a virtual product design based on parameters specified in the SysML model. To establish this process chain, we created a SysML profile, which enables systems engineers to mark blocks of a SysML Block Definition Diagram as «CAD Element» and value properties that should serve as parametric design parameters as «CAD Parameter».

To automate the exchange of model information in the SysML-CAD process chain, we implemented a MagicDraw plug-in that exports and imports model information into a format Autodesk Inventor can process to set the dimensions of a concrete nozzle design automatically.

Since this information exchange only represents the first step of a virtual product development, we applied additional methods from virtual product design such as different simulations to analyze and manufacture a prototype of a concrete Venturi nozzle design.

For this, we analyzed the created Venturi nozzle design by a CFD simulation model based on the Venturi nozzle design specification in the CAD model. By this, we were able to refine the flow simulation results of the Venturi nozzle we already started in the SysML based on parametric diagrams and their analysis. Finally, we used a 3D printer to create a first hardware prototype.

In conclusion, we presented a SysML-CAD process chain and embedded it into a virtual product development environment. For this purpose, we developed a process to connect instances of hardware describing elements in SysML with CAD models, which themselves are integrated into CAx process chains. Moreover, we provided methods and tools to automatically exchange parameters between these instances and an external geometry description of a parametric CAD model in the context of a MDSE methodology.

References

[1] R. France and B. Rumpe, "Model-driven Development of Complex Software: A Research Roadmap," *Future of Software Engineering (FOSE '07)*, pp. 37–54, May 2007.

[2] K. Hölldobler, B. Rumpe, and A. Wortmann, "Software Language Engineering in the Large: Towards Composing and Deriving Languages," *Computer Languages, Systems & Structures*, vol. 54, pp. 386–405, 2018.

[3] A. Levenchuk, "SysML is the Point of Departure for MBSE, Not the Destination," *INCOSE Insight*, vol. 12, no. 4, pp. 54–56, Dec. 2009.

[4] M. Eigner, D. Roubanov, and R. Zafirov, *Modellbasierte virtuelle Produktentwicklung*. Springer Berlin Heidelberg, 2014.

[5] K.-H. Grote, B. Bender, and D. Göhlich, *Dubbel Taschenbuch für den Maschinenbau: Taschenbuch für den Maschinenbau*, 01 2018.

[6] S. Ramo, "Systems engineering manual," *Federal Aviation Agency (FAA)*, 2004.

[7] H. Eisner, *Essentials of project and systems engineering management*. John Wiley & Sons, 2008.

[8] I. C. on Systems Engineering, "Systems Engineering Handbook: A" what To" Guide for All SE Practitioners." INCOSE, 2004.

[9] D. D. Walden, G. J. Roedler, K. Forsberg, R. D. Hamelin, and T. M. Shortell, Eds., *Systems Engineering Handbook: A Guide for System Life Cycle Processes and Activities*, 4th ed. Hoboken, NJ: Wiley, 2015.

[10] A. L. Ramos, J. V. Ferreira, and J. Barceló, "Model-Based Systems Engineering: An Emerging Approach for Modern Systems," *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, vol. 42, no. 1, pp. 101–111, Jan 2012.

[11] *OMG Systems Modeling Language*, Version 1.6 ed., Object Management Group, 2019.

[12] T. V. Huynh and J. S. Osmundson, "A systems engineering methodology for analyzing systems of systems using the systems modeling language (SysML)," *Department of Systems Engineering, Naval Postgraduate School, Monterey*, 2006.

[13] S. C. Spangelo, D. Kaslow, C. Delp, B. Cole, L. Anderson, E. Fosse, B. S. Gilbert, L. Hartman, T. Kahn, and J. Cutler, "Applying model based systems engineering (MBSE) to a standard CubeSat," in *2012 IEEE Aerospace Conference*. IEEE, 2012, pp. 1–20.

[14] *OMG Unified Modeling Language (OMG UML)*, Version 2.5.1 ed., Object Management Group, Dec. 2017.

[15] S. Friedenthal, A. Moore, and R. Steiner, *A Practical Guide to SysML, Third Edition: The Systems Modeling Language*, 3rd ed. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2014.

[16] W. A. Wymore, *Model-Based Systems Engineering*, 1st ed. Boca Raton, FL, USA: CRC Press, Inc., 1993.

[17] R. Koller and N. Kastrup, *Prinziplösungen zur Konstruktion technischer Produkte*. Berlin, Heidelberg: Springer Berlin Heidelberg, 1998.

[18] A. Shah, D. Schaefer, and C. Paredis, "Enabling Multi-View Modeling With SysML Profiles and Model Transformations," 2009, pp. 527–538, the 6th International Conference on Product Lifecycle Management ; Conference date: 06-07-2009 Through 08-07-2009.

[19] A. A. Shah, A. A. Kerzhner, D. Schaefer, and C. J. Paredis, "Multi-view Modeling to Support Embedded Systems Engineering in SysML," *Lecture notes in computer science*, vol. 5765, pp. 580–601, 01 2010.

[20] H. Vangheluwe, J. Lara, and P. Mosterman, "An Introduction to Multi-Paradigm Modelling and Simulation," *Proceedings of the AIS'2002 Conference*, 01 2002.

[21] L. Bassi, C. Secchi, M. Bonfe, and C. Fantuzzi, "A SysML-Based Methodology for Manufacturing Machinery Modeling and Design," *IEEE/ASME Transactions on Mechatronics*, vol. 16, no. 6, pp. 1049–1062, Dec 2011.

[22] L. Wang, M. Izygon, S. Okon, H. Wagner, and L. Garner, "Effort to Accelerate MBSE Adoption and Usage at JSC," *AIAA SPACE 2016*, 09 2016.