Available online at www.sciencedirect.com

ScienceDirect

Procedia CIRP 136 (2025) 420–425

www.elsevier.com/locate/procedia

35th CIRP Design 2025

Development and Implementation of Digital Heterogeneous Models in Model-based Systems Engineering

Thomas Schumacher[a]*, Roman Stephan[a], David Inkermann[a]

[a]*Technische Universität Clausthal, Institute of Mechanical Engineering, Robert-Koch-Straße 32, 38678 Clausthal-Zellerfeld, Germany*

* Corresponding author. Tel.: +49 (0)5323 72 3504. *E-mail address:* schumacher@imw.tu-clausthal.de

Abstract

Model-based Systems Engineering emphasises the use of semi-formal models to describe the overall system behaviour and structure. In addition to these domain-independent models, an efficient engineering process for mechatronic systems requires domain-specific models, like CAD models, to define the detailed design. This contribution introduces heterogeneous models, as product models integrating SysML and CAD model elements, and describes their technical implementation. Therefore, an implementation concept and its technical realisation will be presented. Key elements of this contribution are the realised API to link SysML and CAD tools and its application to develop heterogeneous models.

Keywords: Model-based Systems Engineering; Heterogeneous Models; Model Management; API; SysML; CAD

1. Introduction

The efficient development of modern, mechatronic systems requires a comprehensive system understanding to capture the interactions between various system elements and their interfaces. Using SysML models out of the Model-based Systems Engineering (MBSE) can support this challenge [1]. SysML models have been established over the last few years [2,3] to describe a system's overall structure, behaviour, and parameters [4]. Despite the high importance of SysML models for the development of mechatronic systems, these models will not replace the domain models, like CAD models. These models are still relevant to execute domain-specific engineering tasks, like the definition of the detailed geometrical design or spatial specifications to avoid external influencing factors, like temperature influences. It can be concluded that interdisciplinary engineering tasks, such as the definition of the system architecture, require both kinds of models to enable an efficient engineering process of mechatronic systems. Thereby, the application of different kinds of models within an engineering task can lead to different challenges during the engineering process. The reasons are different model syntax and semantics but also different modelling processes and methods as well as tools [5]. Consequently, failures can occur based on missing information continuity, model inconsistencies or incorrect model understanding, leading to higher engineering costs and time. To avoid these potential failures this publication proposes the application of heterogeneous models. Heterogeneous models enable the integration of different model elements with variable syntax and semantics, into one model visualization [6] based on a linked data structure [7]. The goal of heterogeneous models is to create appropriate models for the respective task and maintain model consistency based on a linked data structure.

2. Research Objective and Questions

The objective of this publication is to present an implementation concept and its technical realisation to create digital heterogeneous models. Thereby, this publication combines SysML and CAD model elements into heterogeneous models.

2212-8271
10.1016/j.procir.2025.08.073

Therefore, the modelling software *FreeCAD* will be enhanced by a specific workbench with additional functions to enable the visualization of heterogeneous models. The linkage of the data structure will be realized by the use of the universal XML data format. The following research questions are addressed in this paper:

- How can digital heterogeneous models be visualised as integrated model representations out of SysML and CAD model elements?
- What is an appropriate implementation concept to ensure model consistency while exchanging model elements between SysML and CAD models?
- What is an appropriate realisation of the introduced implementation concept?

The paper is organised as follows: Section 3 introduces the necessary basics of the state of the art. Section 4 describes the implementation concept and its technical realisation. Section 5 discusses an exemplary application of the realised solution. Finally, the paper concludes with a summary and an outlook on further research.

3. State of the Art

A crucial element of heterogeneous models is the reuse of available knowledge which is stored in different product models. Utilising available knowledge typically requires a model transformation process. This section explains the fundamentals of model reuse and different model transformation approaches.

3.1. Reuse of Models in Engineering Processes

The increasing use of digital models in engineering processes leads to different types of engineering models that are intended to support the execution of engineering tasks [8]. Thereby, models are used as information carriers and knowledge bases about the system [9]. Considering generation-based engineering processes [10], models have great potential for reuse [8], for instance, to support the development of the succeeding product generation. This enables the utilisation of existing system knowledge. The concreteness of the knowledge depends on the type of model used. SysML models contain more abstract knowledge, like the overall architecture and interfaces of the system. Domain models, like CAD models, contain more detailed information about the system, like geometrical or spatial descriptions. Thus, the reuse of models increases engineering efficiency by reducing the engineering effort [11]. The implementation of model reuse requires an appropriate model transformation for the specific engineering processes. Therefore, the following section introduces established model transformation approaches.

3.2. Model Transformation Approaches

The reuse of digital models is often accompanied by a transformation of source models into target models to enable appropriate model reuse. Therefore, Table 1 explains different model transformation approaches. Model transformation is essential to enable model reuse because the initial model is a great knowledge source but is typically not available in the required format. Therefore, models must be transferred into the target format to enable the utilization of available knowledge.

Table 1. Model transformation approaches

Approach	Explanation
Application Programming Interface (API)	An *API* is an application-specific tool which enables a tool to read and interpret data that is normally not in scope. These tool interfaces are mainly realised on the source code level and enable communication between different authoring tools. [12]
Semantical data mapping and meta models	This approach establishes connections between different models or data by using, for instance, semantic web technologies. It involves semantical mapping of the data elements from one model to another [13]. Meta models are often utilised to represent data mapping.
Graph-based data transformation	The application of *graph-based approaches* enables the transformation of specific models to unified model representations. These representations can be transformed into the target model. [14]
Middleware	*Middleware* refers to application-neutral programs that connect applications in a way that their complexity and infrastructure are concealed. [15]

4. Implementation of Heterogeneous Models

This Section introduces the implementation concept of heterogeneous models and describes the technical realisation.

4.1. Implementation Concept

Heterogeneous models shall provide an integrated visualisation based on a linked data structure to avoid model inconsistencies. Therefore, Fig. 1 presents the developed implementation concept. It includes the intended development and integration of an API and its main functions. The programming interface (*API Heterogeneous Models*), highlighted in green, is intended to enable data-based communication between the independent authoring tools *Enterprise Architect* and *FreeCAD*. *Enterprise Architect* is an established SysML modelling tool [16] and *FreeCAD* is a universal tool for CAD modelling. To maintain model consistency, the heterogeneous models shall be generated on a linked data structure. Therefore, the implementation concept intends the enhancement of *FreeCAD* with XMI-based import- and export functions. XMI is an abbreviation for XML Metadata Interchange and is an extension of the XML standard to enable powerful and flexible data exchange

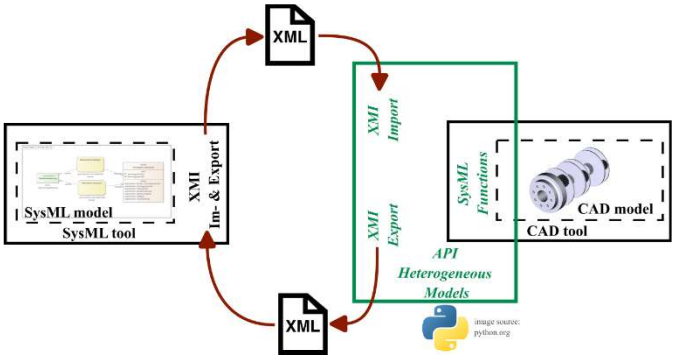

Fig. 1. Implementation concept to interconnect SysML and CAD models

between authoring tools [17]. The use of XML data to exchange data is especially established in SysML modelling tools. However, their intended use is typically the exchange of SysML model elements between different SysML modelling tools and not the data exchange with other kinds of modelling tools. Moreover, the XMI format is a neutral data format that can be read by both humans and machines. Various XMI versions exist today, which specify the data structure of an XML file. In detail, the API to be developed should provide the following main functions:

1. **Visualisation of heterogeneous models.** This API function enables the visualisation of heterogeneous models. For this purpose, *FreeCAD* is to be expanded by an additional workbench, which provides the necessary functions to generate SysML and CAD model elements in one model.
2. **XMI Import.** This function enables FreeCAD to read and interpret XMI-based XML files. To ensure universal applicability, the data structure based on the defined XMI version should not be changed.
3. **XMI Export.** The API should also enable data transfer from the CAD to the SysML tool. Therefore, the implementation concept intends to add an XMI export function to *FreeCAD*. The XML file to be exported must correspond to the specified XMI structure to ensure the data readability without errors.

The API, including the introduced functions, is developed using the *Python* programming language. The following section explains the software-based implementation of the programming interface in detail.

4.2. Technical Realization

The basis for the software-based implementation of heterogeneous models is the identification of a suitable modelling tool that can visualise both SysML and CAD model elements. As SysML tools do not include 3D modelling of objects, heterogeneous models are to be generated with the help of a CAD modelling tool. *FreeCAD* was defined as the modelling tool for heterogeneous models due to its flexible adaptability and expandability. The technical realisation of the introduced main functions of the API is explained below.

4.2.1. Visualisation of Heterogeneous Models

The CAD modelling tool *FreeCAD* offers the option to integrate additional functions and adjust the graphical user interface (GUI) by implementing additional workbenches. Therefore, we developed, with the use of *Python* programming language, an appropriate workbench for the creation of heterogeneous models. The first task was to create and initialize the new workbench named *Heterogene Modelle* for *FreeCAD*. By starting *FreeCAD*, the new workbench will be loaded automatically, see Fig. 2. As the second task, the developed workbench was extended by additional functions to enable the integration of SysML model elements into the heterogeneous model. The following SysML model elements were considered for the heterogeneous model: blocks, requirements, functions, ports and system boundaries. Additionally, the to-be-created heterogeneous model requires CAD model elements, which were integrated into the new workbench from the given CAD modelling functions. These functions primarily contain the creation of drawings and 3D objects, the movement of the model elements within the model and the creation of line and text elements to connect and visualise the relations between the different model elements. All these functions were finally integrated into the

Fig. 2. Initialised workbench *Heterogene Modelle* in *FreeCAD*

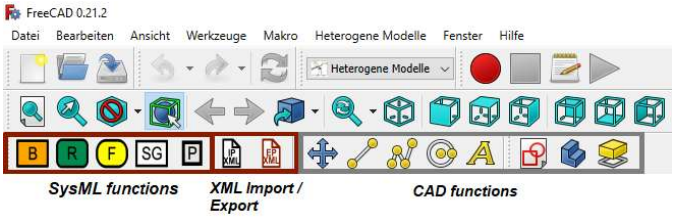

Fig. 3. Graphical user interface within the developed workbench

FreeCAD GUI which is visualised in Fig. 3.

4.2.2. XMI Import

Heterogeneous models shall also support the maintenance of model consistency. Therefore, these models must be created based on a linked data structure. Due to the mentioned advantages in Section 4.1, the data format XML with the XMI data structure (version: 1.1) was defined as the reference. The used SysML modelling tool *Enterprise Architect* can export SysML models as XML files without further adjustments. Therefore, the crucial tasks to enable the XMI import functions into *FreeCAD* were the parsing and interpreting of the XMI structure as well as defining the visualization form of the SysML elements in *FreeCAD*. After analysing the XMI structure a *Python*-based parsing algorithm was developed based on the library *ElementTree [18]*. To realise an XMI import into *FreeCAD* the following functions were developed:

- Read in of model elements and relations,
- Creation of SysML model elements in *FreeCAD,*
- Creation and naming of the connection lines between the model elements.

The read-in function enables an interpretation of the content of the XML file. This function considers in particular the defined SysML model elements and their relations which are required for the creation of heterogeneous models. The second developed function contains the automated creation of the imported model elements in *FreeCAD*. The applied syntax and semantics for the model elements are comparable to the SysML. Finally, the imported relations and their types must be included in *FreeCAD*.

There are two main challenges to overcome when developing the XMI import function. First, the analysis of the XMI structure was challenging because of the actual implementation of the SysML as a profile of the Unified Modeling Language

(UML). For example, many SysML model elements are defined as stereotypes of UML model elements, which leads to a partially confusing XMI structure and impeded parsing. The second challenge was the automated positioning of the imported model elements into the heterogeneous model. In particular, the occurrence of optical overlays had to be avoided using a special algorithm.

4.2.3. XMI Export

Once heterogeneous models have been developed, the model content should be exchangeable with the SysML modelling tool. Based on the implementation concept presented, XML files with the original XMI structure shall be used for this export function. The main objective was to export the model content as an XML file using the XMI format that can be read directly by the SysML modelling tool without additional adjustments. To achieve this objective, the required XMI structure was analysed in more detail to create an XMI reference structure for the export function. This reference structure is organised into six main chapters:

1. *Root elements* which contain mainly the utilized XMI version. In our realisation, we used XMI version 1.1.
2. *Header elements* that define the type and version of the XMI exporter in particular.
3. *Model and package structure*. This XMI part organises the model structure regarding the root model and the to-be-created packages.
4. *Model elements*. After structuring the model using packages, the model elements are included. We included block, requirement, port, and function elements. The main task is considering the required element stereotypes and other important model information. In the XMI structure, this information is defined in the tagged values of the individual model elements.
5. *Relations*. Afterwards, the XMI structure describes the relations between the model elements, distinguishing between dependencies and associations. We considered the relation types *satisfy* and *deriveReqt* as dependencies and *allocate* and *aggregation* as associations. Additionally, all relations describing *item flows* between model elements were typed as associations.
6. *Extensions and closure*. Finally, the XMI structure closes with tool-specific extensions and differences.

The developed *Python* application generates an XML file with the described XMI structure and inserts the concrete model information into the corresponding section.

The second main part of enabling an XMI export function is to realise a read-out function of the required information from the heterogeneous model. Therefore, the developed *Python* API was extended to include the function of searching through all objects in the model and analysing their attributes. The objects are identified using their specific labels and attributes and associated information such as descriptions or directions of relations can be read out. Finally, the developed API shall also be able to extract the created 3D objects out of the heterogeneous model and include them in the XML file as SysML block elements. This function was realised by searching and extracting specific object labels and attributes.

The following section presents an application of the developed *Python* API to create heterogeneous models.

5. Application of Heterogeneous Models

This section presents the application of the introduced heterogeneous models. Previously, the engineering scenario and task, as well as the application example, will be explained.

5.1. Engineering Scenario with Application Example

As engineering scenario, we consider the development of products in generations. This scenario represents the majority of the engineering projects and states that a new product generation leverages on a reference product [10]. This implies that knowledge about the system based on the previous generation is available in the form of models. The engineering scenario aims to reuse model elements in the new product generation to decrease engineering risk. As a product example, we utilise an electromechanical roll stabiliser, which represents a subsystem in the chassis of a car. [19] presents the development of the roll stabiliser in different generations. The hydraulic roll stabiliser is considered as the previous product generation. The main reason for developing a new product generation is changed customer requirements, such as better driving stability, higher working dynamics and better maintainability. Developing the electromechanical roll stabiliser as the new product generation shall satisfy these customer requirements. The considered engineering task comprises developing a heterogeneous model, describing the system architecture of the roll stabiliser based on changed requirements and considering the available knowledge from the previous product generation. To summarise, the task is to develop a heterogeneous model for the system architecture definition in a product generation engineering scenario. The heterogeneous model shall support the system architecture development and analysis due to arising requirement changes.

5.2. Generation of Heterogeneous Models

This section presents the creation of heterogeneous models for the introduced engineering task.

5.2.1. Analysis of Available System Knowledge

Based on the described engineering scenario for the hydraulic roll stabiliser both SysML and CAD models already exist. The first task is to analyse these models enabling an appropriate knowledge reuse. Fig. 4 shows parts of the SysML model which shall be imported into the heterogeneous model. Additionally, CAD models are available as STEP files from previous product generations. Both models are directly readable using the developed API. Fig. 5 visualises the resulting initial heterogeneous model after importing the XML and STEP files. The resulting heterogeneous model provides a first basis for analysing the existing model elements and their relations as well as deciding which elements can be reused for the new roll stabiliser generation.

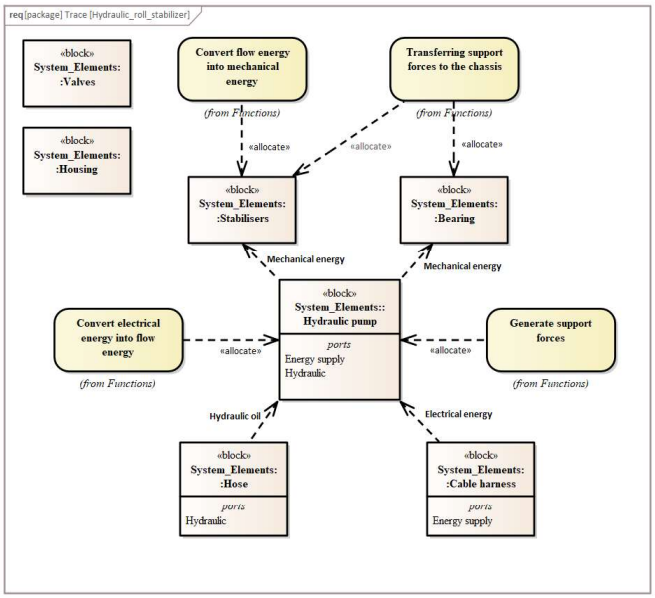

Fig. 4. SysML partial model for the hydraulic roll stabiliser

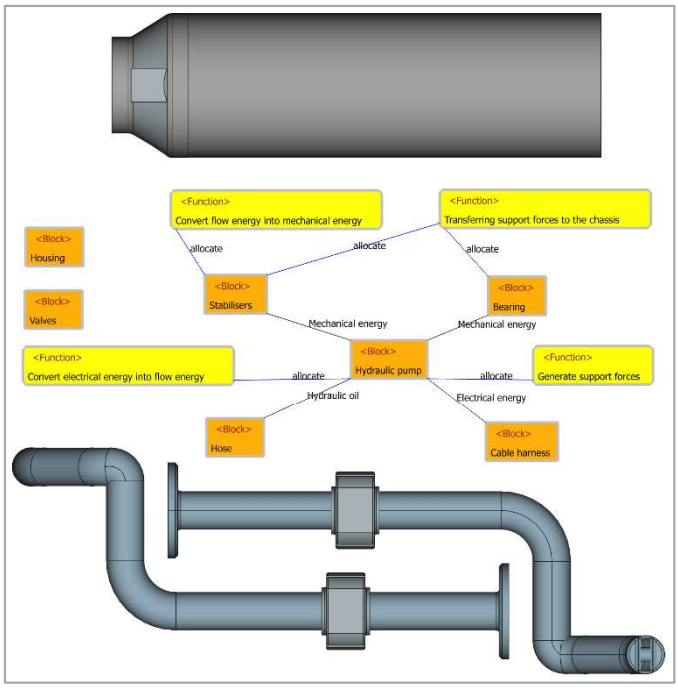

Fig. 5. Heterogeneous model after import of SysML and CAD model elements

5.2.2. Investigation of Changed Requirements

After importing available model elements and defining which model elements can be reused from the previous product generation the model will be extended by the changed requirements. This activity shall support identifying the affected functions or blocks based on the changed requirements.

5.2.3. Development of System Architecture

After investigating the changed requirements and the available knowledge from the previous product generation, the system architecture will be developed to satisfy the change requirements. Therefore, additional model elements will extend the initial heterogeneous model, and the different model element types will be linked. Fig. 6 presents an exemplary system architecture to satisfy these changed requirements. For better model readability, all imported SysML model elements are marked with a grey frame while all newly developed elements are marked with a black frame. Furthermore, the relations and item flows are differentiated by colour. Relations are marked as blue and item flows as black lines.

5.2.4. Updating initial SysML model

After finalising the system architecture development based on the changed requirements, the initial SysML model shall also be updated. The added system elements and functions must be reimported into the original SysML model. A special interest is the transformation of the developed 3D objects into SysML block elements. Fig. 7 presents the resulting SysML model after the transformation of the heterogeneous model in the SysML model by applying the developed export function.

A comparison of these two models could imagine that the SysML model does not contain the defined interface descriptions (ports) out of the heterogeneous model. However, they are included in the SysML model but not part of the visualisation because the applied diagram type does not allow the visualisation of ports at the system boundary.

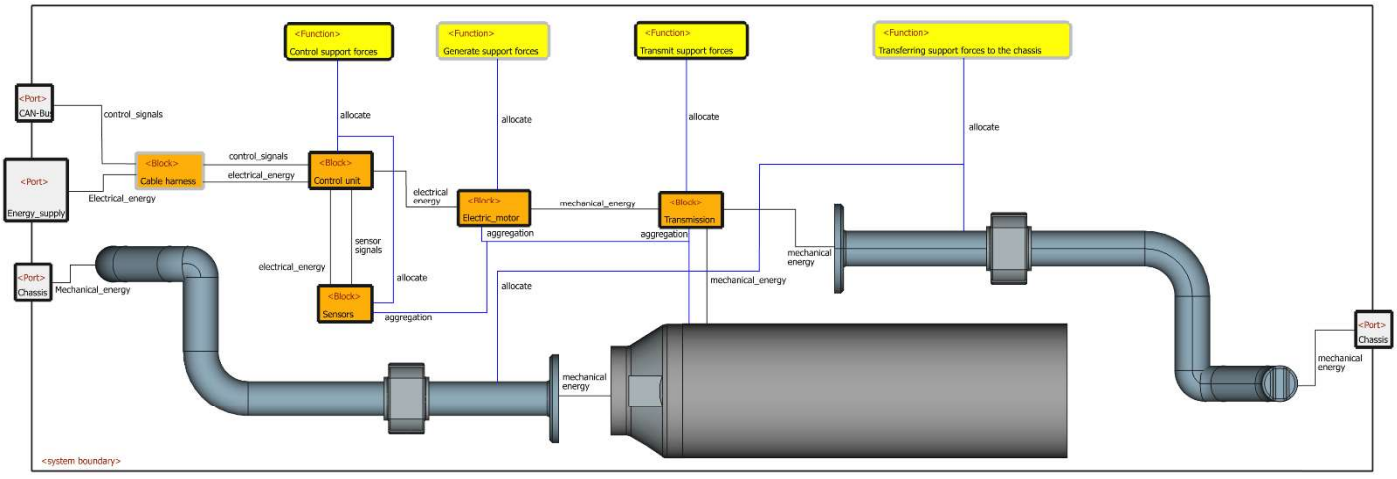

Fig. 6. Heterogeneous model of electromechanical roll stabiliser

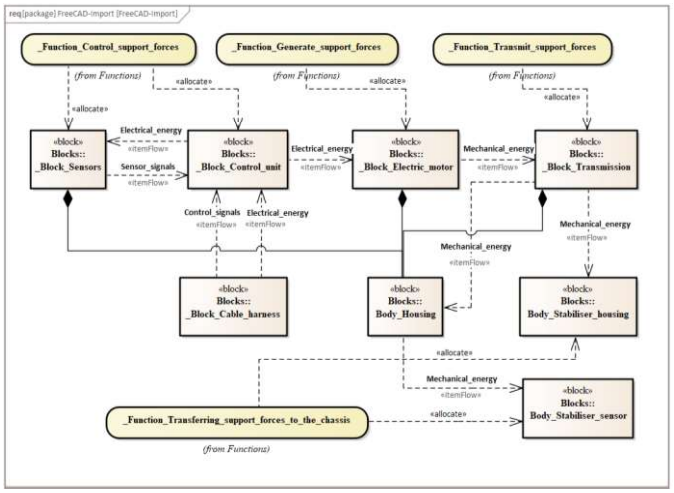

Fig.7. SysML model after reimport

6. Summary and Outlook

Applying different model types in an interdisciplinary engineering task can lead to various failures because of missing information continuity, model inconsistencies or incorrect model understanding. To address this challenge this publication proposes the application of heterogeneous models.

Heterogeneous models integrate model elements with varying syntax and semantics, into one model based on a linked data structure. Within this publication, the development and technical implementation of heterogeneous models is described. We focus on the integration of SysML models as domain-independent models and CAD models as an exemplary domain model. First, this publication presents an implementation concept which describes the required functionalities to link the SysML and CAD structure by the use of XML files. Afterwards, the technical implementation in the form of a *Python*-based programming interface (API) and its integration into a CAD modelling tool as a specific workbench is explained. The main tasks in the realisation of the API were the connection of the underlying data structure, which was realised by using export and import functions based on the XMI data structure, and the extension of the CAD tool with SysML-specific functions. Finally, the application of the realised API to create heterogeneous models was demonstrated by using a specific application example. Future work will focus on the evaluation of heterogeneous models with students and practitioners.

Acknowledgements

Funded by the Lower Saxony Ministry of Science and Culture under grant number ZN3493 within the Lower Saxony “Vorab“ of the Volkswagen Foundation and supported by the Center for Digital Innovations (ZDIN).

References

[1] Henderson K, Salado A. Value and benefits of model ‐ based systems engineering (MBSE): Evidence from the literature. Systems Engineering 2021;24(1):51 ‐ 66. https://doi.org/10.1002/sys.21566.

[2] Gausemeier J, Dumitrescu R, Steffen D, Czaja A, Wiederkehr O, Tschirner C. Systems Engineering in Industrial Practise. Paderborn: Heinz Nixdorf Institut, Universität Paderborn, Lehrstuhl für Produktentstehung, Fraunhofer-Institut für Produktionstechnologie IPT ‐ Projektgruppe Entwurfstechnik Mechatronik, UNITY AG; 2015.

[3] Berschik MC, Schumacher T, Laukotka FN, Krause D, Inkermann D. MBSE within the Engineering Design Community – an Exploratory Study. Proc. Des. Soc. 2023;3:2595–604. https://doi.org/10.1017/pds.2023.260.

[4] Friedenthal S. A practical guide to SysML: The systems modeling language. Waltham, MA: Morgan Kaufman; 2014.

[5] Gausemeier J, Schäfer W, Greenyer J, Kahl S, Pook S, Rieke J. Management of Cross-Domain Model Consistency during the Development of Advanced Mechatronic Systems. In: International Conference on Engineering Design, ICED’09. Stanford, CA, USA; 2009, p. 1–12.

[6] Jansen S, Welp EG. A Heterogeneous Modelling Approach for Domain Allocation in Mechatronics. Guidelines for a Decision Support Method Adapted to NPD Processes 2007.

[7] Schumacher T, Inkermann D. Heterogeneous models to Support Interdisciplinary Engineering - Mapping Model Elements of SysML and CAD. Procedia CIRP 2022;109:653–8. https://doi.org/10.1016/j.procir.2022.05.309.

[8] Tao F, Ma X, Liu W, Zhang C. Digital Engineering: State-of-the-art and perspectives. Digital Engineering 2024;1:100007. https://doi.org/10.1016/j.dte.2024.100007.

[9] Mordecai Y, Dori D. Towards a Quantitative Framework for Evaluating the Expressive Power of Conceptual System Models. INCOSE International Symp 2016;26(1):42–57. https://doi.org/10.1002/j.2334-5837.2016.00144.x.

[10] Albert Albers, Nikola Bursac, Jan Urbanec, Robert Lüdcke, Galina Rachenkova. Knowledge Management in Product Generation Development – an empirical study. Unpublished; 2014.

[11] Trujillo A, Weck OL de, Madni AM. An MBSE Approach Supporting Technical Inheritance and Design Reuse Decisions. In: ASCEND 2020. Reston, Virginia: American Institute of Aeronautics and Astronautics; 2020.

[12] Shinde RS, Rathod S, Ekar T, Bhongade S, Pachpore SS. Development of API for Estimating Torsional Strength of Shaft through Knowledge-Based Engineering Approach. International Research Journal of Engineering and Technology (IRJET) 2020:1209–14.

[13] Shalom G. The Role of Semantic Web Technologies in Improving Knowledge Management Systems. EJIKM 2024;3(1):26–37. https://doi.org/10.47941/ejikm.1751.

[14] Song G, Zhang K, Kong J. Model Management Through Graph Transformation. In: 2004 IEEE Symposium on Visual Languages - Human Centric Computing. IEEE; 2004, p. 75–82.

[15] Meyer M, Weingärtner S. Enterprise Application Integration — Grundlagen. In: Meyer M, editor. CRM-Systeme mit EAI. Wiesbaden: Vieweg+Teubner Verlag; 2002, p. 199–229.

[16] Ma J, Wang G, Lu J, Vangheluwe H, Kiritsis D, Yan Y. Systematic Literature Review of MBSE Tool-Chains. Applied Sciences 2022;12(7):3431. https://doi.org/10.3390/app12073431.

[17] Grose TJ, Doney GC, Brodsky SA. Mastering XMI: Java programming with XMI, XML, and UML. New York, Weinheim: Wiley; 2002.

[18] xml.etree.ElementTree — The ElementTree XML API. [August 22, 2024]; Available from: https://docs.python.org/3/library/xml.etree.elementtree.html.

[19] Schumacher T, Inkermann D. Investigation of advantages of models and the modelling process by introducing a model evaluation concept. Proc. Des. Soc. 2024;4:2735–44. https://doi.org/10.1017/pds.2024.276.